Sapienza - Università di Roma

DOCTORAL THESIS

Operations Research: from Computational Biology to Sensor Network

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy in

Operations Research

Author:
Maria De Cola

Supervisor:
Dr. Giovanni Felici

Faculty of Information Engineering, Computing and Statistics
Department of Statistical Science

Rome, 2013

To my little hero Jacopo

Abstract

In this dissertation we discuss the deployment of combinatorial optimization methods for modeling and solve real life problem, with a particular emphasis to two biological problems arising from a common scenario: the reconstruction of the three-dimensional shape of a biological molecule from Nuclear Magnetic Resonance (NMR) data.
The first topic is the 3D assignment pathway problem (APP) for a RNA molecule. We prove that APP is NP-hard, and show a formulation of it based on edge-colored graphs. Taking into account that interactions between consecutive nuclei in the NMR spectrum are different according to the type of residue along the RNA chain, each color in the graph represents a type of interaction. Thus, we can represent the sequence of interactions as the problem of finding a longest (hamiltonian) path under the constraint that the edges of the path follow a given order of colors (the orderly colored longest path). We introduce three alternative IP reformulations of APP obtained with a max flow problem on a directed graph with packing constraints over the partitions, which have been compared among themselves. Since the last two models work on cyclic graphs, for them we propose an algorithm based on the solution of their relaxation combined with the separation of cycle inequalities in a Branch & Cut scheme.
The second topic is the discretizable distance geometry problem (DDGP), which is a formulation on discrete search space of the well-known distance geometry problem (DGP). The DGP consists in seeking the embedding in R^k of a undirected graph given a set of Euclidean distances between certain pairs of vertices. DGP has two important applications: (i) finding the three dimensional conformation of a molecule from a subset of interatomic distances, called Molecular Distance Geometry Problem, and (ii) the Sensor Network Localization Problem. We describe a Branch & Prune (BP) algorithm tailored for this problem, and two versions of it solving the DDGP both in protein modeling and in sensor networks localization frameworks. BP is an exact and exhaustive combinatorial algorithm that examines all the valid embeddings of a given weighted graph $G = (V, E, d)$, under the hypothesis of existence of a given order on V. By comparing the two version of BP to well-known algorithms we are able to prove the efficiency of BP in both contexts, provided that the order imposed on V is maintained.

Acknowledgements

I wish to thank many people. First of all, my supervisor Giovanni Felici. An amazing mentor both from the professional and the personal point of view. He led me through this Ph.D. like a big brother teaches to his sister to ride a bike: he supported my desire to climb into the saddle of this Ph.D., he gave me a lot of useful advices, he pushed me a little bit when I was stuck, he left me go alone hoping for my best, he reprimanded me when I swerved, he helped me to get up when I fell encouraging me to climb back on the bike. He gave me the possibility to work on interesting topics with other high level researchers as Leo Liberti, Antonio Mucherino, Carlile Lavor, Marta Szachniuk, Jacek Blazewicz, just to name a few, and allowed me to present my results in many places around the world. One of these is Rennes, where I spent few months during my third year working side by side with Antonio Mucherino. This experience was very important, both for my professional and personal life. Thanks to Alessio and Filippo for having been my family there.

I have spent most of my Ph.D. time working at the Institute of Systems Analysis and Computer Science "A. Ruberti" of the Italian National Research Council (IASI-CNR). There, in addition to my supervisor, I have met many people who have supported my research and at the same time have been friends. First of all the director Paola Bertolazzi, who I thanks for her generosity and scientific support, but also Emanuel, Guido, Daniele, Marianna, and other friends.

Special thanks goes to my Ph.D. program coordinators, Paolo Dell'Olmo and Stefano Lucidi, and to the Department of Statistics at "Sapienza-University of Rome". I also wish to thanks Adriana Cirenei, whose professionalism and helpfulness make her a pillar of this Ph.D. program.

Thanks to Rosella for her support on the study of the NMR spectroscopy. Chiara and Serena for their help on the final review. Alessandro, Francesca, Gloria, Alessandra and the Lucio Sestio's friends for their friendship demonstrated during these years in Rome. Caterina and Eleonora for being a constant in my life. Thanks also to Christina Leslie that gave me the opportunity to visit her lab at the Sloan-Kettering Institute, and to Livia, Chiara, Nicola, Phaedra and Julie for made me feel at home also in NY. Finally, I would like to thank my mother for her support, patience and love, and my son Jacopo for the strength he gives me everyday.

Contents

List of Figures

List of Tables

It is good to have an end to journey toward;
but it is the journey that matters,
in the end.

Ursula K. Le Guin

In God we trust. All others must bring data.

W. Edward Deming

Chapter 1

Introduction

1.1 OR meets biology

Making the best decision in any situation is one of the most coveted man desires. Since the efforts of military planners during the World War II [84], Operations Research (OR) is known as the discipline that helps to make better decisions. In these last 70 years the OR applications have reached areas from business, industry, logistics to the newest ecology, social network and biology, to name few. This rapid growth is due to least two factors. The first one is the substantial progress made in the OR techniques improvement, such as the simplex method for solving linear programming problems, developed by George Dantzing in 1947 and listed in [53] among the top 10 algorithms of the twentieth century. The second factor that plays a key role in the OR growth is the arrival of the computer revolution. In fact, the development of personal computers and good OR software packages from the 80s makes possible nowadays to collect and to handle a huge amount of data. Nevertheless, for years OR has been a little known science to who is outside the profession. With the goal of making understandable the value of OR even outside the OR scientific community, in 2004 the Institute for Operations Research and the Management Sciences (INFORMS) introduced the "O.R.: The Science of Better" campaign to "improve the visibility, identity, and image to key constituencies outside the discipline"[1].

In the last decades the research carried out to solve more and more complex problems has followed two main directions: first, an improvement of the solvers and algorithms, taking also into account the increasing power of computers. Second, the way to model problems. These two aspects are in fact two sides of the same coin, since a good solution of an optimization problem is obtained by means of both an appropriate model (also

[1] www.scienceofbetter.org

called formulation) and an efficient algorithm to solve it. More precisely, the process which leads from a real-world problem to its solution by means of OR can be resumed in the following 4 steps:

1. formalize the (real-world) problem;
2. create an abstract mathematical model to describe the problem;
3. give the model as input to a solver in order to obtain the optimal solution (if the solution process is too much time and/or memory demanding due to the difficulty of the problem, and the optimal solution cannot be found, usually the solver can provide some other information as the best solution found so far and sometimes a bound on the cost of the optimal solution);
4. interpret the solution within the real-world setting of the problem.

Modeling suitably the problem is as important as the use of an efficient solver for identifying a solution. In fact, the model directly affects solvers performance and the possibility to map the optimal solution into the real-world domain. The recent explosion of data generated in biology and medicine opens the doors to models and solution methodologies inspired by biological processes. High throughput biological data need to be processed, analyzed, and interpreted to address problems in life sciences. Whole genome sequencing and other molecular diagnostics, as well as magnetic resonance spectroscopy and imaging have brought light on the molecular mechanisms of evolution of diseases, increasing the importance in the healthcare value chain as powerful platforms for precision diagnosis and selection of optimal treatment. In this scenario, OR based methods are taking place with excellent results in the language of optimization, stochastic processes and graph theory, and in many cases they have become the workhorses of the research. In particular, the ability of OR techniques of analyzing complex molecular problems makes OR one of the most successful branches of applied mathematics for such problems. This should not be surprising if we consider the huge cost of an in vitro experiment compared to computer simulations. Moreover, there exists a bilateral relation between a biological experiment and an OR method: the first one provides the data to guide the development of the model and its analysis, whereas the latter is useful to draw the next experimental designs.

If we look at the recent literature, we can find many ties between biology and operational research, including sequence alignment [23], single nucleotide polymorphisms and haploids [20, 87], assembling DNA segments [23, 50], gene expression [125, 138], and protein folding [67], which is perhaps the most celebrated problem in computational biology. In [13] optimization methods are used for topics ranging from model building and optimal

experimental design to metabolic engineering and synthetic biology, while [88] and [137] collect most of the optimization algorithms developed by the machine learning community with application in computational system biology. A more complete overview about both theoretical and computational methods in bioinformatics and system biology is given in [42]. The book includes the description of linear programming techniques developed to reconstruct gene regulatory networks, transcriptional regulatory networks, protein interaction networks, and metabolic networks.

Based on the above description, the goal of this dissertation is to show how much powerful are the OR techniques for modeling and solving real-world problems, with a particular attention to the OR contributions in Computational Biology. We will describe two different mathematical programming approaches for modeling and solving two biological problems that arise from a common scenario: the reconstruction of the three-dimensional shape of a biological molecule from Nuclear Magnetic Resonance (NMR) data.

The remainder of this first chapter is organized as follows: in Section 1.2 we introduce the scenario in which the problems described in this dissertation take place. Finally, in Section 1.3 we summarize the main scientific contributions of the thesis.

1.2 Scenario

The biological function of a molecule is determined by its specific folding into a three-dimensional structure, defined by the atomic coordinates. This particular shape is called tertiary structure. Although the primary structure (see Section 2.4.1) of a molecule is its template for folding, since it contains the information that specifies both the tertiary structure and the pathway to obtain that shape, it is not true that identical primary structures always fold similarly. Conformations differ based on environmental factors as well (i.e., based on where they are found). Several neurodegenerative and other diseases are believed to result from misfolded molecules. For example, the incorrect folding of certain proteins inhibits the production of antibodies for many allergies [121].

For years researchers have focused only on DNA and protein structure determination. Only recently the research has also been extended to RNA, which is implicated in all aspects of the genetic regulation, as described in Section 2.4.1.1. Understanding the functions of RNA and proteins is essential to know their three-dimensional structures, which, due to various technical reasons, are very difficult to determine [18].
The development of many analytic methods has made possible to obtain some indirect structural data on which the tertiary structures may be determined. For example, the diffraction data for a molecule crystal can be obtained by X-ray crystallography and

used to find the electron density distribution and hence the structure of the molecule; the magnetic resonance spectra of the nuclear spins in a molecule can be detected by nuclear magnetic resonance (NMR) spectroscopy and used to estimate the distances between certain pairs of atoms and subsequently, the coordinates of the atoms in the molecule. In either case, a set of experimental data is collected and a mathematical problem needs to be solved to form the structure [119]. The are some advantages in using NMR spectroscopy are that the molecule does not need to be crystallized (which is time-consuming and often fails), NMR determines an unique fold of the molecule, and NMR also provides dynamic proprieties of the molecule such as the flexibilities of the backbone or protein sidechains [38]. For these reasons, NMR is a powerful tool for the analysis of folding transitions in RNA and proteins. However, to obtain accurate enough signals, NMR experiments can only be carried out for small molecules with less than a few hundred residues.

The determination of a three-dimensional shape of a molecule by NMR data is a lengthy and complicated process [129]. Although types of NMR experiments differ for proteins and nucleic acids [142], all methods of NMR structure analysis follow the same sequence of stages: data acquisition and processing, peak picking, assignment, derivation of spatial restraints, structure calculation, and validation [70, 124, 129].
The first stage concerns the acquisition of the multi-dimensional correlation spectra, which will be computationally analyzed in the next steps. The sample is prepared by a chemical synthesis, placed in a probe, and inserted in NMR spectrometer. Next a range of one- and multidimensional NMR experiments are executed on it. All obtained spectra are recorded. A more detailed description of the operating principle of NMR spectroscopy is provided in Section 2.4.2.
During the *peak picking* phase, the frequencies of all peaks from the spectrum are extracted, and their values are either stored in a tabular form (i.e. in a peak table) or graphically displayed over the spectrum. Figure 1.1 shows a traditional peak picking on an one-dimensional spectrum. The spectrum represents the resonance signals of the protons of the circled molecule in top-right, where the numbers upon each peak are the resonance frequencies (chemical shift) of the corresponding protons.
The output of any peak picking algorithm is a plain list of significant points in a spectrum, each representing a nucleus involved in the experiment and provided of their chemical shift values. However, the peaks are not labeled or marked according to their type, they are just peaks without a semantic framework. An experienced spectroscopist can often identify crucial peaks with virtual certainty and, if necessary, make an assignment on the basis of a single, uniquely identified peak [70]. On the contrary, when automatic analysis is in progress this lack of information has some important consequences on the whole process of structure determination. Therefore, the spectrum needs to be further

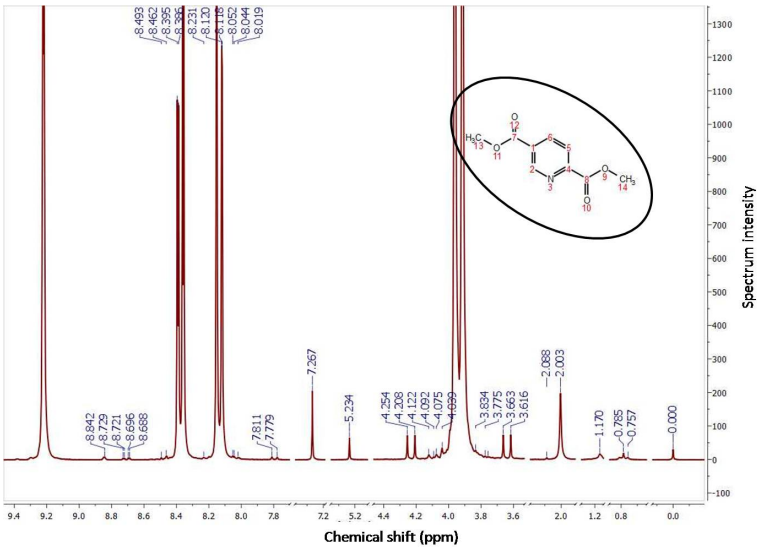


FIGURE 1.1: Illustration of traditional peak picking (the picture is taken from www.nmr-analysis.blogspot.com).

processed so that appropriate nuclei are assigned to each pick in order to identify the resonance signal yielded by the NMR experiment. This identification is provided by the *assignment* stage, achieved by a *sequential walking* that uses information derived from NMR experiments, such as chemical shift values and the number of peaks of each individual nucleus (i.e., multiple resonance signals). The sequential walking process correlates any cross-peak (see Section 3.2.1 for the definition of cross-peak) to the correct nucleus "walking" across the NMR spectrum. This pathway can be used to generate a draft of the structure, although for a more precise reconstruction other parameters must be calculated. There exist several software for automating the assignment stage (such as NVR-BIP [8], CANDID [77], ARIA [98], FLYA [120],to name a few). Unfortunately, these programs can be only applied to protein spectra. Therefore, the development of new procedures assigning to appropriate nuclei the corresponding signals is crucial for improving the state of the art of RNA structural analysis. Here takes place the first topic described in Chapter 3. If we consider a graph whose vertices represent the nuclei and the edges are colored according to the resonance signals (for example, taking into account the specificity of the required connectivity between consecutive nuclei signals in the NMR spectrum), then the reconstruction of the sequence of interactions is equivalent to finding the longest path on the graph such that the edges of the path must follow a given order of colors.

Once resonance signals have been identified, it is possible to calculate the *structural restraints*, based on chemical shifts obtained thought the sequence-specific assignments. A set of possible restraints include: molecular distances (e.g., distances between hydrogen atoms), torsion angles (dihedral angles around certain bonds), coupling constants, etc.

[142]. In general, the more restraints are used the better is the precision of the structures generated. In fact, these structural restraints can be used as input for the *structure calculation* process, during which an ensemble of structures of the molecule is determined. The most popular methods for structure generation are distance geometry (DG) methods developed in order to satisfy the covalent and structural restraints [90]. The atoms coordinates provide a structure that can be considered an approximation to the real one [74]. This structure can be refined using optimization, e.g., by an energy minimization procedure with the distance ranges as structural restraints. Here takes place the second topic described in Chapter 4. If we consider a graph whose vertices represent atoms and the edge weights are Euclidean distances between pair of atoms, which are known *a priori* from a subset of interatomic distances, then determining the three-dimensional conformation of the molecule is equivalent to determine the coordinates of the atoms, by solving a graph embedding problem [26]. A more general and abstract form of the DG problem is to find the coordinates for a set of points in some topological space given the distances between certain pairs of points. Therefore, in addition to protein modeling, the problem has applications in many other fields as well, such as graph drawing [73] and wireless sensor networks localization [10], to name a few.
The final step in NMR structure determination is the validation of the set of structures identified. The aim is to obtain an indication of the quality and structural statistics, such as a measure of the fit of the structures to be experimental data, and other scores [124]. Validation helps to understand whether an unique final conformation of the molecule may be real or resulting from errors, and as such whether the restraint should be modified, i.e., whether false assignment were made or whether the bounds of the restraints should be adjusted. Nevertheless, validation is often only used for structural statistics calculation, such as a description of the resulting structure [124].

It is legitimate to point out that although the two problems described in this dissertation are two steps of the same process, they have been studied separately, and developed for different biomolecules, RNA (RiboNucleic Acid) and proteins, respectively.

1.3 Contributions

In this dissertation two combinatorial problems related to the molecular structure determination are presented. Each problem takes place in a specific step within the process of determination of the three-dimensional folding of the molecule using NMR data (see Section 1.2). As regards the first topic, we improve the formulation of the assignment pathway problem proposed in [129] through three alternative models, and we prove the NP-hardness of this problem. The second topic, instead, concerns the development of

a new version of a combinatorial algorithm, originally drawn for molecular instances [94, 97], able to solve sensor network instances.

The dissertation is organized as follows. In Chapter 2 we introduce basic concepts of mathematical programming, of computational complexity theory, and some fundamental definitions of graph theory used throughout the thesis. We also provide a brief account of the needed biological background.

In Chapter 3 we introduce the 3D assignment pathway problem from three-dimensional Nuclear Magnetic Resonance (NMR) map of a RNA molecule. We prove that this problem is NP-hard, and show a formulation based on edge-colored graphs. Taking into account that interactions between consecutive nuclei in the NMR spectrum are different according to the type of residue along the RNA chain, each color in the graph represents a type of interaction. Thus, we can represent the sequence of interactions as the problem of finding a longest (hamiltonian) path under the constraint that the edges of the path follow a given order of colors. We consider three alternative IP models formulated by means of max flow problems on a directed graph with packing constraints over certain partitions of the vertices for reformulating the problem. Since the last two models work on cyclic graphs, for them we propose an algorithm based on the solution of their relaxation combined with the separation of cycle inequalities in a Branch & Cut scheme. Part of the chapter is taken from [49] and it is to be published in [127, 128].

In Chapter 4 we describe the discretizable distance geometry problem (DDGP), which is a formulation on discrete search space of the well-known distance geometry problem (DGP) related to protein modeling and sensor networks localization. We give a brief review of the existing continuous approaches to the solution of the DGP, and then we show a few combinatorial requirements nedeeded to reduce the search space from continuous to discrete. Finally, we describe the Branch & Prune (BP) algorithm, and two versions of it solving the DDGP both in protein modeling and in sensor networks localization frameworks. BP is an exact and exhaustive combinatorial algorithm that examines all the valid embeddings of a given weighted graph $G = (V, E, d)$, under the hypothesis of existence of a given order on V. By comparing these two version to well-known algorithms, we prove the efficiency of BP in both contexts, provided that it is held the order imposed on V. Part of this chapter is taken from [51].

In Chapter 5 we summarize the work. We describe the current work in progress, discussing the potential impact of our contributes, especially in NMR molecular modeling. We conclude the chapter by discussing some important issues for future investigations. Finally, since the topics of this dissertation touch concepts of several branches of science, in addition to Chapter 2 we provided two appendices, in order to give to the reader the basis for fully understanding the sense of the problems described in this dissertation.

Chapter 2

Background

In this chapter we introduce basic concepts of several branches of science, in order to give to the reader the basis for fully understanding the sense of the problems described in this dissertation. At the beginning we rough in mathematical programming (MP), and three big classes of MP problems. In fact, the problems presented in this dissertation belong to two of these three class of problems. We briefly introduce the computational complexity theory, and some fundamental definitions of graph theory used throughout the thesis. Finally, there is a section devoted to molecular biology and NMR spectroscopy, since the biological nature of the dissertation.

2.1 Mathematical programming

Mathematical Programming (MP) is a branch of OR which can be employed to analyze and solve real-world problems where one wants to maximize, or minimize, an objective function subject to some constraints on the decision variables. A detailed introduction of MP is beyond the scope of this section. For a full introduction to the MP theory we suggest the books [21, 110, 140]. Nevertheless, in a more precise definition we can express a generic MP formulation as:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in X, \end{aligned} \tag{2.1}$$

where X is the set of feasible solutions (also called *search space* or *feasible region*), which is a cartesian product of continuous and discrete intervals (as it is defined by the constraints of the problem and the bounds on the variables), and $f : X \to \mathbb{R}^F$ represents the set of $|F|$ objective functions (if $|F| > 1$ we have a multiobjective problem; in this thesis we always consider problems where $|F| = 1$). The problem represented by the

model (2.1) can be expressed as: find a point $x^* \in X$ (called *optimal solution* or *global optimum*) which minimizes the objective function $f(x)$, that is $\forall x \in X, f(x^*) \leq f(x)$. A point $\bar{x} \in X$ is called *local optimum* if exists $\epsilon > 0$ such that $\forall x \in X, ||x - \bar{x}|| \leq \epsilon$ and $f(\bar{x}) \leq f(x)$, i.e., there are not better solutions than $f(\bar{x})$ in the neighborhood of $\bar{x}$. If a problem does not admit any optimal solution, it is called *infeasible* problem, that is $X = \emptyset$. Note that in the rest of this chapter we always refer to minimization problems. A maximization problem where one wants to maximize an objective function f can be reformulated as a minimization problem by means of the relationship max f = -min $-f$.

If the search space X is *convex*[1], then the set of global optima is the same as the set of local optima. Intuitively, they are easier to solve, since there is no need to continue the search for a global optimum after having found a local optimum, whilst in general this is not true.

2.1.1 Classification of MP problems

We can now propose a classification of the MP problems formulated in the very general form (2.1). Remember that the set X is given by the bounds and kinds (as integer, continuous, or discrete) of the variables, and by the constraints of the problem, which are usually on the form $g(x) \leq 0$ or $h(x) = 0$. There are several classifications of MP problems, depending on the degree of the objective function and the constraints as well as the nature of the variables. Here we classify the MP problems in three main classes:

- Linear Programming (LP): the objective function and the constraints are linear, and the variables are continuous;
- Mixed Integer Linear Programming (MILP or MIP): the objective function and the constraints are linear, and at least one variable is integer. If all variables are integer, Integer Linear Programming (ILP or IP) is used in place of MILP to refer to the problem;
- Nonlinear Programming (NLP): at least one among the objective function and the constraints is nonlinear, and the variables are continuous[2];

We can further write the following relationships: LP $\subset$ MILP and LP $\subset$ NLP. The meaning is that if a solver can be employed for a given class of problems Π, then it can also be employed for problems of all the classes $\Pi \subset \Pi'$. For instance, a MILP solver can

[1] In an Euclidean space, a set X is convex if for every pair $x, y \in X$, every point z on the straight line segment that joins x and y is also included in X.

[2] When at least one variable is integer the problem is referred as Mixed Integer Nonlinear Programming (MINLP).

be employed to solve a LP problem as well as a LP solver can be used to solve a MILP instance. But the latter will ignore the integrality constraints on the variables. These relationships give also an intuitive idea about the complexity of the problems of the different categories. In general $\Pi \subset \Pi'$ means that Π is easier to solve than Π'. Hence, LP problems are usually the easiest to solve, whereas NLPs are the most difficult. It is possible to go further into detail with the categorization of MP problems, but for this thesis the previous classification suffices.

The assignment problem presented in Chapter 3 is formulated by three IP models. In Chapter 4 the distance geometry problem is a NLP problem, whose objective function involves bilinear or quadrilinear terms depending on which formulation is used.

2.1.1.1 Linear programming

In a LP problem the constraints and the objective function are linear. In its standard form, a LP problem can be expressed as:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0, \end{aligned}$$

where c^T is the n dimensional row vector of coefficients for the objective function, A is the $m \times n$ matrix constraints, b is the m dimensional column vector representing the right-hand side of the constraints, and x is the n dimensional column vector of the nonnegative variables of the problem. The feasible region of such a problem is a convex set called convex polyhedron, having a finite number of vertices. If the polyhedron is bounded it is called *polytope*. The importance of this concept in LP is that the optimal solution of a LP problem corresponds to a vertex of the polytope representing the feasible region. This has been the key observation at the base of the simplex algorithm, that is an algorithm which starts from a vertex of the polyhedron and moves to another adjacent vertex as long as the objective function improves. The procedure stops when the vertex representing the optimal solution is reached. This is the main idea, but a lot of details are missed (e.g., how to perform this move from a vertex to a better one, how to know if the optimal vertex is found). For more information, see [48]. Although this algorithm has an exponential complexity in the worst case, it is efficient in practice.

Many LP solvers,e.g. CPLEX [80], implement the simplex's method. LPs are important because a lot of real-world problems can be described in this way. Moreover, LPs arise during the solution process of other categories of MP problems, as for example MILPs.

2.1.1.2 Mixed integer linear programming

A MILP problem consists of a linear objective function and some linear constraints, where a subset of the variables are integer. In general solving a MILP problem is NP-hard [65]. However, there is a special case where the optimal solution of a MILP problem can be obtained by relaxing the integrality constraints and solving the resulting LP problem (called continuous relaxation). Consider the MILP problem stated in the standard form as follows:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \in X \\ & \forall i \in I \quad x_i \in \mathbb{Z}, \end{aligned} \tag{2.2}$$

where I is the set of indices of integer variables. Let us introduce the concept of unimodularity taken from [61]:

Definition 2.1. A $m \times n$ matrix A, where $m \leq n$, is called *unimodular* if for all $m \times m$ submatrices B of A it holds that $det(B) \in \{-1, 0, 1\}$.

Suppose that the polyhedron defined by (2.2) is not empty and limited (i.e., it is a polytope). Then the Theorem 2.1 from [61] holds.

Theorem 2.1. *Let the $m \times n$ matrix A be unimodular and the m dimensional column vector b be integer valued. The polyhedron associated to (2.2) has only integer vertices.*

It is known that the optimal solution of a LP problem is found on a vertex of the polyhedron defined by the constraints of the problem. If we relax the integrality constraints of the MILP problem, and solve the corresponding LP produces an integer solution, then this solution is optimal for the MILP problem. In other words, the unimodularity of the constraint matrix A together with the integrality of the components of the vector b is a sufficient condition for obtaining the optimal solution of the MILP problem by solving its continuous LP relaxation. In the case of problems where the constraints $Ax = b$ are casted in form of inequalities, the concept of unimodularity has to be substituted with that of *total unimodularity* in order to preserve the property of having integer vertices of the polyhedron (the difference with respect to the unimodularity of Definition 2.1 is that, for total unimodularity, the property $detB \in \{-1, 0, 1\}$ must hold for all $m \times m$ square submatrices B of A).

2.1.1.3 Non linear programming

Nonlinear problems can be defined as follows:

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & \forall i \in M & g_i(x) \leq 0 \\ & x \in X & \end{array} \tag{2.3}$$

where $M = \{1, \ldots, m\}$ and at least one among $g_i(x)$ and $f(x)$ is a nonlinear function. If there are no constraints on the variable, the problem is called unconstrained. Finding the optimal solution of a NLP problem is not as easy as for LP and MILP, due to the nonlinearities and in general nonconvexities (in this case could occur several local optima which makes the search for the global optimum by the solver difficult).
There exist some necessary conditions for the optimality called Karush-Kuhn-Tucker (KKT) [83, 86], which must be satisfied by a solution x^* of a NLP problem to be a local optimum, and which are used by some NLP solvers. They can stated as follows.

The KKT conditions
Given a NLP problem in the form (2.3), a feasible point $x^* \geq 0$ which respects some regularity conditions is a local optimum only if there exist some multipliers $\mu_i, \forall i \in M$ such that following conditions hold:

$$\begin{array}{lll} \forall i \in M & g_i(x^*) \leq 0 & (primal\, feasibility) \\ \mu_i \geq 0 & & (dual\, feasibility) \\ \nabla f(x^*) + \sum_{i=1}^{m} \mu_i \nabla g_i(x^*) = 0 & & (stationarity) \\ \forall i \in M & \mu_i g_i(x^*) = 0 & (complementary\, slackness), \end{array}$$

where the objective function f and the constraints g_i are differentiable in x^* and the operator ∇ applied to a function express its gradient. Some of the most common regularity conditions are called Linearly Independent Constraint Qualifications (LICQ) and require the gradient of the constraints that are active at x^* to be linearly independent when evaluated at x^*.

2.2 Graph theory

The problems described in this dissertation are modeled on graphs. A *graph* is a mathematical object which represents a binary relationship on a set of elements.
In this section we give the fundamental definitions of graph theory, and some notation

used throughout this thesis. Some others will be given later when necessary. For an overview on the general theory of graphs, the reader can refer to one of these books [3, 27, 29].

An *undirected* graph $G = (V, E)$ consists of a set V of *vertices* (or nodes) and a set E of *edges* whose elements are unordered pairs of distinct vertices, i.e., for all edge $(i, j) \in E$ holds that $(i, j) = (j, i)$. If both V and E are finite, then G is a *finite* graph.
An edge (i, j) is *incident* to vertices i and j, which are called its *endpoints*. Vertices $i, j \in V$ are adjacent if the edge (i, j) belong to E. A *loop* is an edge whose endpoints are the same vertices. A graph that has no loops and no more than one edge connecting a pair of vertices is a *simple* graph. If there exists an edge connecting any pair of vertices, then the graph is *complete*. A graph is *weighted* if a number (weight) is assigned to each edge. Such weight might represent a cost, length, etc. depending on the problem at hand. Given a graph $G = (V, E)$, the graph $G' = (V', E')$ is *subgraph* of G if (i) $V' \subseteq V$ and (ii) $E' \subseteq E$. When a graph G' is a complete subgraph of another graph G then it is a *clique*. Given a subset of vertices $U \subseteq V$ of G, the subgraph $G[U] = (U, E')$ is *induced* by U in G if (i) $U \subseteq V$ and (ii) $E' = \{\{u, v\} \in E | u, v \in U\}$. Similarly, given a subset of edges $S \subseteq E$ of G, the subgraph $G[S] = (V', S)$ is *induced* by S in G if (i) $S \subseteq E$ and (ii) $V' = \{v \in V | \{u, v\} \in S\}$. Note that in the first case the subgraph is induced by a subset of vertices, whereas in the second one the subgraph is induced by a subset of edges.
A *directed* graph (or digraph) $D = (V, A)$ consists of a set V of *vertices* (or nodes) and a set A of *arcs* whose elements are ordered pairs of distinct vertices, i.e., for all edge $(i, j) \in A$ holds that $(i, j) \neq (j, i)$. Throughout this thesis we refer to a directed weighted graph as *network*.

2.2.1 Paths and cycles

Given a finite undirected simple graph $G = (V, E)$ where $|V| = n$ and $|E| = m$.
A *walk* between two vertices in G is a sequence of edges $\{e_0, e_1, \ldots, e_k\}$ such that $e_i = (v_i, v_{i+1})$, for $i = 0, \ldots, k$. The vertex v_0 is called *source*, whereas the last vertex v_k is called *destination*. A *trail* between two vertices in G is a walk without repeated edges. A *path* between two vertices in G is a walk without repeated vertices. The *length* of a path is the number of edges that compose it. If there exists in G a path from any pair of vertices, then G is a *connected* graph. A *cycle* in G is a path closed, i.e. a path where source and destination are the same vertex. A path traversing all the vertices of G is a *Hamiltonian path*. A cycle in G traversing all vertices is a *Hamiltonian cycle*.

2.3 Computational complexity

This section serves as an introduction to the areas of complexity theory needed for this thesis. For a more deep introduction to complexity theory, the reader is referred to [65].

2.3.1 Algorithm complexity

In general, one is interested in solving combinatorial problems as efficiently as possible, where efficient usually means fast. Hence, an important criterion for the classification of problems is the time the best known algorithms need to find a solution for the given problem. This issue is addressed by the theory of computational complexity. Its main purpose is to classify problems according to their difficulty to be solved by any known algorithm. For the classification of problems it has been shown to be useful to address the question regarding problem complexity as a worst-case measure, that is, the complexity of a problem is determined by the hardest conceivable instance.

The time-complexity of an algorithm is measured by a time-complexity function that gives, depending on the instance size, the maximal run-time for the algorithm to solve an instance. The size of a problem instance reflects the amount of data to encode an instance in a *compact* form. Often it is sufficient to have an intuitive understanding of the size of an instance. The time-complexity is typically given, in terms of the number of elementary operations like value assignments or comparisons; it is formalized by the $O(\cdot)$ notation. Let f and g be two functions from $\mathbb{N} \to \mathbb{N}$, then we write $f(n) = O(g(n))$ if there are positive integers c and n_0 such that for all $n > n_0$, $f(n) \leq cg(n)$.

An algorithm runs in *polynomial* time, if the worst case run-time is bounded by a polynomial; otherwise the algorithm is said to be an *exponential* time algorithm.
In complexity theory, a basic difference is made between efficiently solvable problems (*easy* problems) and inherently intractable ones (*hard* problems). Usually, a problem is considered efficiently solvable if a solution can be found in a number of steps bounded by a polynomial of the input size. If the number of steps needed to solve an instance grows super-polynomially, we say that a problem is inherently intractable.

2.3.2 Problem Complexity

The theory of NP-completeness formalizes the distinction between easy and hard problems. In general, the theory of NP-completeness is concerned with the decision version of combinatorial problems. The generality of the conclusions drawn is not limited by this fact because it is obvious that the optimization version of a problem is not easier

to solve than the decision version and if the optimization version of a problem can be solved efficiently, then the same is true for the decision version. Optimization problems typically have an associated decision problem; for example, in the Traveling Salesman problem (TSP), the associated decision version asks whether a tour with cost bound $f(\pi) < L$ exists [65]. The evaluation version of an optimization problem can be solved as a series of decision problems using binary search on the bound L.
The theory of NP-completeness distinguishes between two basic classes of problems. One is the class P of tractable problems.

Definition 2.2. The class P is the class of decision problems that can be solved by a polynomial time algorithm.

The class NP can be defined informally in terms of a *nondeterministic* algorithm. Such an algorithm can be conceived as being composed of a *guessing* stage and a *checking* stage. If we are given some instance I, in the first stage some solution is guessed. This solution is verified by a deterministic polynomial algorithm in the second stage. The class NP is the class of problems that can be solved by such a nondeterministic algorithm. For the class NP this polynomial-time verifiability of the property for some given solution s is essential. The polynomial time verifiability also implies that the guessed solution is of polynomial size.

Definition 2.3. The class NP consists of those problems that can be solved by a nondeterministic polynomial-time algorithm.

Any decision problem that can be solved by a deterministic polynomial-time algorithm also can be solved by a nondeterministic polynomial-time algorithm, that is P $\subseteq$ NP.
Probably the most important open question in theoretical computer science today is whether P = NP? It is widely believed nowadays that P $\neq$ NP, yet no proof of this conjecture has been found so far.
A problem usually is considered intractable if it is in NP $\setminus$ P. As one cannot show that NP $\setminus$ P is not empty, the theory of NP-completeness focuses on proving results of the weaker form if P $\neq$ NP, then $\Pi \in$ NP $\setminus$ P. One of the key ideas needed for this approach is the notion of polynomial-time reducibility among problems.

Definition 2.4. A problem Π is polynomially-reducible to a problem Π', if a polynomial-time algorithm exists that maps each instance of Π onto an instance of Π' and that for each instance of Π "yes" is output iff for the corresponding instance of Π' the output of the decision procedure is "yes".

Informally this definition says that if Π can be polynomially reduced to Π', then problem Π' is at least as difficult to solve as problem Π. Using the notion of polynomial reducibility we can proceed to define the class of NP-complete problems.

Definition 2.5. A problem Π is NP-complete iff (i) $\Pi \in$ NP and (ii) for all $\Pi' \in$ NP holds that Π' is polynomially reducible to Π.

The class of NP-complete problems is in some sense the class of the hardest problems in NP. If a NP-complete problem can be solved by a polynomial time algorithm, then all problems in NP can be solved in polynomial time. Yet, so far for no NP-complete problem a polynomial time algorithm could be found. Thus, if one can prove that a problem Π is NP-complete common belief suggests that no deterministic polynomial-time algorithm exists and the problem cannot be solved efficiently. The foundations of NP-completeness theory were laid in [44]. Cook gave the first proof that every problem in the class NP could be polynomially reduced to the satisfability problem (SAT).

In this dissertation we are concerned with optimization problems and address the search version of the problem, that is, we want to find optimal solutions. Clearly, the search version is not easier than the associated decision problem. Thus, proving that the decision version of a problem is NP-complete implies that also the search version is hard to solve. Problems which are at least as hard as NP-complete problems but not necessarily element of NP are called NP-hard.

Definition 2.6. A problem Π is NP-hard iff for all $\Pi' \in$ NP holds that Π' is polynomially reducible to Π.

Therefore, any NP-complete problem is also NP-hard. On the other side, if the decision version of an optimization problem is NP-complete, the optimization problem is NP-hard.

2.4 Biomolecules and NMR spectroscopy

In this section we introduce basic concepts of molecular biology and trace out the explanation of the NMR spectroscopy process, in order to make more understandable the importance of the problems discussed through this thesis and the available data used.

2.4.1 Biomolecules

The fundamental biomolecules are carbohydrates, lipids, nucleic acids, and proteins. *Carbohydrates* are commonly referred to as sugars, and they perform numerous roles in living organisms, the most famous being the storage of energy. The *lipids* conduct many biological functions acting as structural components of cell membranes. *Nucleic acids*, which include DNA (deoxyribonucleic acid) and RNA (ribonucleic acid), and *proteins*

function in encoding, transmitting and expressing genetic information. These are the most important biomolecules in a cell. In particular, proteins perform a vast array of functions including replicating DNA, responding to stimuli, and transporting molecules from one location to another.

Nucleic acids are made from *nucleotides* (or *residue*) successively linked each other by chemical bonds (*backbone* structure). Each nucleotide is composed of a *nucleobase* (e.g., adenine, cytosine, guanine, thymine or uracil), a sugar with five carbon atoms, and one or more phosphate groups. Therefore, a *nucleotide chain* is a backbone of sugar and phosphate on which are bounded the nucleobases. Nucleic acids differ in the structure of the sugar in their nucleotides, and in one type of nucleobase (adenine, cytosine, and guanine are found in both RNA and DNA, while thymine occurs in DNA and uracil occurs in RNA). The nucleotide chain represents the genetic information needed to define all proteins in a cell through a particular process called protein synthesis. This process transforms the sequence of nucleotides into a well-defined sequence of *amino acids*, which in turn compose the proteins. Such sequences are called *primary structures*: a nucleotides chain is the primary structure of a nucleic acid, and an amino acid chain is the primary structure of a protein. The *secondary structure* describes one- and two-strand fragments as well as the formation of loops or helices in local segments of the molecule, while the *tertiary structure* is its whole folding on itself in a three-dimensional structure, defined by the atomic coordinates. This specific three-dimensional structure of the molecule (both for Nucleic acids and proteins) determines its activity. A full complete biochemical description of these biomolecules is beyond the scope of this thesis. The reader interested to a more biochemical detailed description is referred to any book of biochemistry as [18].

2.4.1.1 Biomelules in the origin of life

DNA, RNA and proteins are the building blocks of life. DNA is the famous molecule of the heredity. This is the molecule that gets passed down from one generation to the next. Proteins are the molecules of structure and functions. Cells are packed full of proteins. RNA is the intermediary between DNA and proteins. Hence DNA molecules code for the protein molecules by RNA molecules making us what we are. This process, known as "central dogma of molecular biology", has been introduced in 1958 by Francis Crick. He stated hat information passes from DNA to proteins via RNA, but proteins cannot pass the information back to DNA [46]. This information transfers describe the normal flow of biological information: DNA can be copied to DNA (DNA replication), DNA information can be copied into mRNA[3] (transcription), and proteins can be synthesized

[3]mRNA is the messenger RNA, that is a copy of DNA that convey genetic information.

using the information in mRNA as a template (translation). The biological dogma laid the foundations for the study of residue-by-residue information transfers in biomolecules where one chain (e.g. amino acids chain) is used as a template for the construction of another chain (e.g. protein) with a sequence that is entirely dependent on the original one.

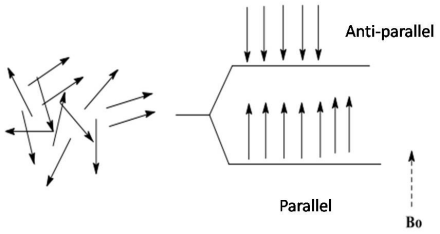
Since DNA acts as genetic information repository of all the cells, and proteins perform a wide array of functions dictated by the nucleotide sequence of the genes, initially the research in sequence information area was focused on these two biomolecules. Only recently the research has been extended to RNA, which is implicated in all aspects of the genetic regulation, such as the control of both transcriptional and post-transcriptional gene expression. In fact, in eukaryotes the 97% of the transcriptional output is not employed for coding the DNA (ncRNA), hence it is not translated into protein. This discover implied a dramatic increasing of studies aimed at understanding the importance of RNA in many biological processes. In particular, the discoveries of ncRNA and RNA interference[4] (RNAi) have involved a broad line of research. Although most of ncRNA is a component of the ribosome on which mRNA carries out the protein synthesis, recently some transcriptomic and bioinformatics studies have suggested the existence of thousands of small ncRNAs involved in the post-transcriptional gene silencing, the mechanism of inhibition the translation of a particular gene. Such small ncRNA, discovered at the end of the 90s, can be placed within the nucleus as well as in the cytoplasm, and their length ranges from 20-30 nucleotides. They are known as small interfering RNA (siRNA) and MicroRNA (miRNA), and are the shortest RNA in eukaryotes. Usually, these small ncRNA recognize homologue sequences within mRNA and through based-pair complementary sequence induct the degradation of the RNA target or block the protein synthesis process [16]. Over the last few years, the dysregulation of miRNA has been associated with disease, first of all the cancer [75, 104], for that miRNAs ar often called *oncomir*.

In 1986 the Nobel laureate Walter Gilbert introduced the phrase “RNA world” in a commentary on how recent observations of the catalytic proprieties of various forms of RNA fit with this hypothesis [66]. The discovery of RNA catalytic capability, in fact, supported the newer hypothesis that an RNA world existed on Earth before modern cell arise. According to this hypothesis, RNA stored both genetic information and catalyzed the chemical reactions in primitive cells. Only later in evolutionary time did DNA take over as the genetic material and proteins become the major catalyst and structural component of cells [4].

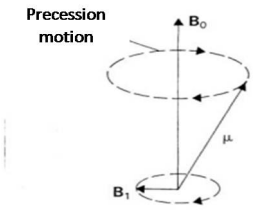
[4]RNA interference is a biological process in which RNA molecules inhibit gene expression, typically by causing the destruction of specific mRNA molecules.

2.4.2 Nuclear Magnetic Resonance Spectroscopy

Nuclear Magnetic Resonance (NMR) Spectroscopy is an analytical technique that allows to obtain detailed information on the molecular structure of chemical compounds. NMR Spectroscopy measures the absorption of electromagnetic radiation in molecules immersed in a strong magnetic field [78]. A molecule is formed by *atoms*, which in turn consist of a central *nucleus* containing a mix of positively charged *protons* and electrically neutral *neutrons*, surrounded by a cloud of negatively charged *electrons*. An atom containing an equal number of protons and electrons is electrically neutral, otherwise it is positively or negatively charged. An atom is classified according to the number of protons and neutrons in its nucleus: the number of protons determines the chemical element, and the number of neutrons determines the *isotope* of the element. Therefore, isotopes are variants of a particular chemical element such that they share the same number of protons in each atom, but differ in neutron numbers. For example, carbon-12, carbon-13 and carbon-14 are three isotopes of the element carbon each with 6 protons and with 6, 7 and 8 neutrons respectively. Any subatomic particle orbits around a own axes generating a small magnetic field. This rotation motion is called *spin*. In particular, the rotation motion of the whole nucleus (protons and neutrons) around its own axes is called *nuclear spin*, and the magnetic field generated by it is said nuclear magnetic moment (NMM) of spin. All isotopes that contain an odd number of protons and/or neutrons have a nonzero NMM of spin, while all nuclei with even numbers of both have a NMM spin of zero. Only nuclei with nonzero NMM of spin can absorb and re-emit the applied electromagnetic radiation, and are thus observable in an NMR experiment. In absence of a external magnetic field, the NMM associated to the nuclear spin assumes any direction in the space. But, when the nuclei with NMM are placed in an external static magnetic field, the nuclei are split into two potential energy levels (spin states) assuming a parallel or anti-parallel orientation with respect to the applied magnetic field (see Figure 2.1(a)). In these spin states the nuclei also undergo a cone shaped rotation motion called *precession*. This motion looks like the motion of a spinning top (see Figure 2.1(b)). The precession frequency of a spin, called Larmor frequency (LF), depends on the nucleus under investigation as well as on the chemical environment. I.e., two atoms of different chemical elements have different LF, and two atoms of the same chemical elements (e.g. two isotopes) have different LF if they are not surrounded by the same chemical structure. In presence of the static magnetic field, the resonant frequency signal (i.e., the LF of spin) can induce a transition between spin states since the frequency of radiation is equivalent to the energy difference between the two levels. When it occurs some spins switch from parallel to anti-parallel orientation at higher energy state (spin flip), and it is said that the nuclei *resonate*. When the radio frequency signal is then switched off, the relaxation of the



(a) parallel and anti-parallel orientation. B_0 is the external magnetic field direction.



(b) B_0 is the external magnetic field direction, and B_1 is the direction of the MMN of spin that follows the precession.

FIGURE 2.1: Parallel and anti-parallel orientation (a) and precession motion (b).

spins backing to the lower state produces a measurable amount of radio frequency signal at the resonant frequency associated with the spin flip. The exact frequency of the signal produced during the spin relaxation is specific for each type of atom and depends on the local chemical environment. The energy absorbed by the nuclei is slowly transferred to the neighboring atoms with an intensity proportional to the number of nuclei able to resonate during the radiation. In other words, a NMR experiment involves only nuclei of atoms of specific chemical elements of the analyzed molecule, which yield resonance signals depending on the other atoms surrounding them. The collected signal during the NMR experiment, called *correlation* signal, is an oscillating signal with the LF of the nucleus in examination, which fades over time and it is called FID (Free Induction Decay). The FID is then transformed in a function of frequency, the NMR spectrum, by Fourier transformation[5] (see Figure 2.2). As above mentioned, when an atom is placed

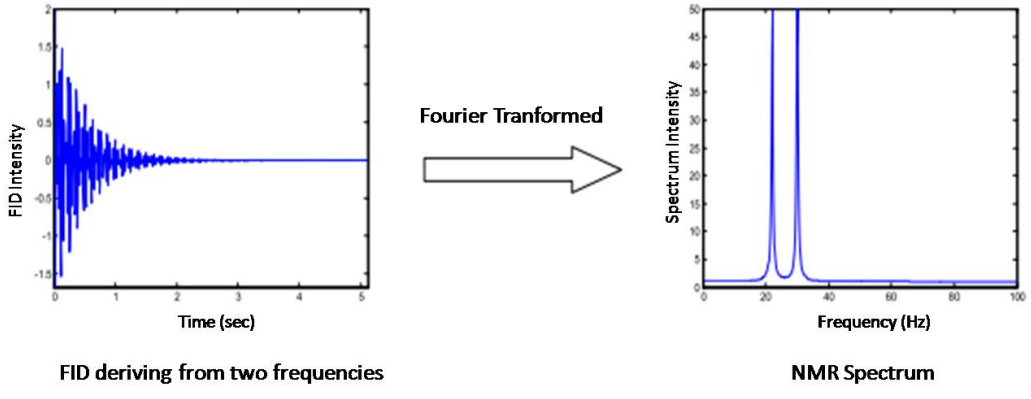


FIGURE 2.2: Conversion from a FID graph to a NMR spectrum.

in a magnetic field, its electrons circulate around the direction of the applied magnetic

[5]The Fourier transform converts a time function into a function of frequency, and it is one of the most common approaches in signal analysis [30].

field. This circulation causes in turn a small magnetic field, which shields the nuclei slightly from the external field. This induce a variation of LF, since the electron density around each nucleus in a molecule varies according to the types of atoms and bonds in the molecule. This effect is called *chemical shift*, and it gives the name to the relative signal frequency in a NMR spectrum. The chemical shift value is reported in parts per million (ppm), and it is one of the major parameters of NMR spectroscopy since it causes the different positions of the signals in a NMR spectrum. For organic structure determination, the two most important types of NMR spectra are proton ($H1$ NMR) and carbon ($C13$ NMR) spectra. They give information about the number of hydrogens and carbons in a molecule and their connections. NMR spectroscopy experiments can be carried out by one- (1D), two- (2D), three- (3D) and four-dimensional (4D) techniques. All 1D NMR spectroscopy experiments are carried out using a single pulse sequence. For example, a 90° pulse (applied along the x axis) rotates the magnetization vector onto the y axis. After this pulse each spin processes with its own LF around the z axis and induces a signal in the receiver coil of the NMR spectrometer. Usually, the experiment is repeated several times and the data are summed up to increase the signal to noise ratio. After summation the data are Fourier transformed to yield the final 1D spectrum. In additional to the 1D NMR spectroscopy used to study chemical bonds, two or three dimensional approaches have been developed for the determination of the structure of complex molecules like proteins and nucleic acids. In 2D NMR spectroscopy experiments, in addition to preparation and detection steps which define 1D NMR spectroscopy experiments, the spins can process freely for a given time t_1 and is used a mixing sequence. In particular, the nuclei are excited with two pulses or groups of pulses. The acquisition is carried out by incrementing the delay (i.e. evolution time t_1) between the two pulse groups. Two dimensional Fourier transformed yields the 2D spectrum with two frequency axes. 2D NMR spectroscopy includes homonuclear and heteronuclear correlation experiments. In homonuclear experiments, signals produced by the same isotope (usually $H1$) are detected. These signals can be produced by the atoms being in close relation through bond or through space. The 2D COSY (Correlated Spectroscopy correlates scalarly coupled protons) and 2D TOCSY (Total Correlation Spectroscopy identifies protons belonging to the same scalar coupling network) experiments correlate all atoms which are in close relation through bond. The 2D NOESY (Nuclear Overhauser Effect Spectroscopy) experiment correlates all protons which are close enough. It also correlates protons which are distant in the amino acid sequence but close in space due to tertiary structure. This is the most important information for the determination of protein structures. Note that an homonuclear experiment involves just protons of hydrogens nucleus (which does not contain neutrons), and only in this case proton and nucleus are synonymous. Heteronuclear experiments detect the signals generated by different isotopes. The most important heteronuclear NMR experiment is the HSQC

(Heteronuclear Single Quantum Correlation). It correlates the nitrogen atom of an NHx group with the directly attached proton. Each signal in a HSQC spectrum represents a proton that is bound to a nitrogen atom. A 3D NMR spectroscopy experiments can be easily constructed from a two dimensional one by inserting an additional indirect evolution time and a second mixing period between the first mixing period and the direct data acquisition. Each of the different indirect time periods (t_1, t_2) is incremented separately. A 3D NMR spectroscopy experiment can be achieved by combining HSQC and NOESY in a single 3D experiment: The NOESY experiment is extended by an HSQC step. Acquisition starts after this HSQC step rather than at the end of the NOESY mixing time. The resulting experiment is called 3D NOESY-HSQC. In a similar way, a TOCSY-HSQC can be constructed by combining the TOCSY and the HSQC experiment. A 3D NMR spectroscopy experiment can also be constructed by triple resonance experiments. These experiments are called *triple resonance* because nuclei of three different chemical elements (e.g. $H4', C4', P$) or three protons are correlated. Figure 2.3 shows a 3D NMR spectrum representing the correlations between three protons. Any dimension corresponds to the chemical shift ranges of the specific nucleus involved in the NMR experiment; each point on the spectra is a resonance peak, and its volume is proportional to the signal intensity.

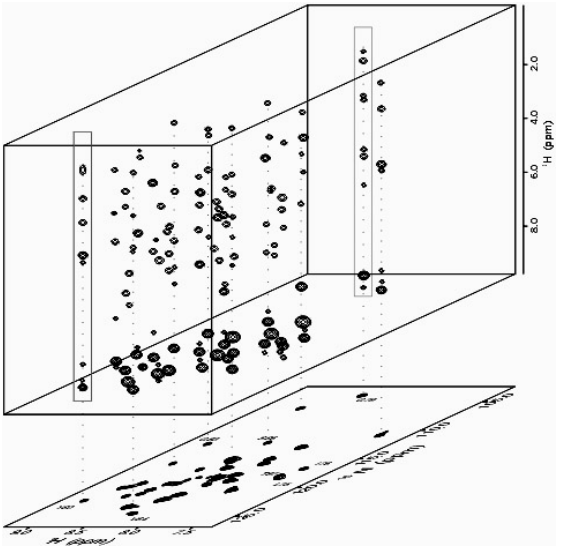


FIGURE 2.3: Absorption spectrum of a 3D NMR experiment (the picture is taken from http://www.bioc.rice.edu/bios576/nmr/nmr.html).

Chapter 3

The orderly colored longest path problem for RNA structure determination

In this chapter we undertake the discussion on the 3D assignment pathway problem from three-dimensional NMR map of a RNA molecule. The idea, originally proposed in [129], is that determining the sequence of interactions among atoms involved in the NMR experiment can lead to determine the shape of such biological molecule. First, we introduce the assignment pathway problem in RNA structural analysis, we prove that this problem is NP-hard, and show a formulation of it based on edge-colored graphs. Taking into account that interactions between consecutive proton in the NMR spectrum are different according to the type of residue along the RNA chain, each color in the graph represents a type of interaction. Thus, we can represent the sequence of interactions as the problem of finding a longest (hamiltonian) path under the constraint that the edges of the path follow a given order of colors. The problem is referred to as the Orderly Colored Longest Path on a c-edge-colored graph (OCLP). Next, we consider three alternative IP models formulated by means of max flow problems on a directed graph with packing constraints over certain partitions of the vertices [49]. Since the last two models work on cyclic graphs, for them we propose an algorithm based on the solution of their relaxation combined with the separation of cycle inequalities in a Branch & Cut scheme [127]. By means of an ad-hoc generator, we assemble a large set of simulated test problems that respect the structural features of real 3D NMR maps recorded for RNA molecules. Using these problems as a benchmark, we provide computational evidence of the utility of our models as well as of the algorithm based on cycle separation. This chapter is mainly based on the works presented in [49, 127]. However, [129] was a guide for the introduction to the biochemical problem and for its graph-based modeling.

3.1 Introduction

In the last years a quick growth of operations research techniques used for modeling and solving many molecular processes has been observed. Understanding the functions of each macromolecule in the cell has made the cognition of organism structure one of the most fundamental tasks in many different research areas. At the beginning the researches focused on DNA and proteins. However, at present we can observe a growing interest in RNA study, which has been the subject of a multitude of recent discoveries, including the involvement of regulatory RNAs in cancer [16] as well as infectious and neurodegenerative diseases [62]. The knowledge of three-dimensional structure of RNA is essential to understand its biological functions, including the identification of conformational changes that accompany its folding. High-resolution Nuclear Magnetic Resonance (NMR) spectroscopy can provide both structural details and dynamic characteristics of biomolecules without requiring crystallization, which is impossible to achieve in some samples. This makes of NMR spectroscopy a powerful tool for the analysis of folding transitions in RNA. Fundamental element of the analysis is an identification of resonance signals among nuclei in the molecule analyzed. An assignment of the observed NMR signals to the corresponding nuclei is a bottleneck of the RNA structure elucidation [129]. Hereafter we refer to this problem as the *assignment pathway problem*. The number of correlation signals recorded during a NMR experiment grows with the molecule size, producing spectra with more and more overlapping peaks. This high density arrests or disables resonance signal identification on the basis of two-dimensional experiments. A step towards three-dimensional spectra is the most evident solution to this problem [129].

The procedure based on our idea for solving the *assignment pathway problem* on 3D NMR maps (further also referred to as 3D-APP) consists in three phases. The first involves the building of an edge-colored graph $G = (V, E)$ from a 3D NMR spectrum such that any vertex $v \in V$ is a cross-peak (see Section 3.2.1), and the edges are colored according to the type of interaction occurring in the NMR experiment between consecutive nuclei along the RNA chain. These interactions can be computed by using the coordinates of each cross-peak, i.e., using the chemical shift values of the nuclei. The second phase concerns to find a longest (hamiltonian) path on G in order to respect the sequence of iterations, which is considered as a sequence of colors on G. Thus the magnetization transfer pathway (or the correlation signals pathway) occurring between the nuclei is represented through an orderly colored path (see Section 3.3.1 for the formal definition). The last phase provides the assignment of the corresponding atoms[1] to each

[1]Since the NMR spectroscopy involves only nuclei of the atoms (see Section 2.4.2), in this framework when we refer to atom we mean nucleus and vice-versa.

cross-peak according to the type of NMR experiment adopted.
To our knowledge, this is the first time where an assignment pathway problem is formulated as a longest (hamiltonian) path on an edge-colored graph. Unfortunately, at moment not many experimental data for already solved cases are collected and publicly available for an analysis. This because of the 3D NMR for RNA has been introduced relatively late, as compared to the protein field. Thus, we implemented an instance generator to simulate the reliable 3D NMR data together with the assignment solution. This drawback brought us to focus especially on the second stage of this procedure for which we developed and compared three integer linear programming models to solve the orderly colored longest path (OCLP) problem in order to reconstruct the magnetization transfer pathway.

The chapter is structured as follows. In the remainder of this Section we introduce the importance of the assignment pathway problem for RNA structure determination. In Section 3.2 we describe the biophysical origin of the pathway reconstruction problem on 3D NMR spectra and we discuss its computational complexity. In Section 3.3 we define the OCLP problem, by showing how an instance of this problem can represent an instance of the 3D-APP as well as an instance of different kinds of well-known problems. In Section 3.4 we describe three alternative integer programming models for modeling in turn the OCLP problem as a network problem, and we show this category of problems still being NP-hard whilst formulated on an acyclic network. Section 3.5.1 introduces the procedures proposed for solving the three models. In Section 3.6 we describe two instance generators used to (i) generate edge-colored graphs, and (ii) simulate spectral data. In this way we have compared the models both from the mathematical and the biochemical point of view. We also evaluate the efficacy of the fractional cycle separation approach as opposed to the separation of the integer cycles in the optimal solutions of the problem. Finally, in Section 3.7, we draw conclusions and show the directions of future work.

3.1.1 The problem of the assignment in RNA structural analysis

Elucidating the mechanistic aspects of many cellular processes requires a detailed knowledge of the tertiary structure of the RNA molecules involved. This folding into a specific three-dimensional shape depends on the primary and secondary structures of the RNA molecules (see Section 2.4.2). In fact, RNA usually possesses a variety of single-stranded and double-stranded regions that give rise to complex three-dimensional structures.
Differently from protein and DNA, which have slower degradation under in vitro conditions, the development of methods dedicated to the exploration of RNA structure is ambler.

As mentioned in Section 1.2, the two main experimental techniques used to derive structure models of biological biomolecules are X-ray crystallography and Nuclear Magnetic Resonance spectroscopy. X-ray crystallography (also referred as X-ray diffraction) outdoes NMR in the resolution of experimental data. Nevertheless, the crystallization of several RNA molecules is not possible, it does not reflect the molecular dynamics, and often it is not reproduce the real conformation of the molecules [124, 129]. High resolution NMR study can provide both structural details and dynamic characteristics of the molecule. In this respect, NMR spectroscopy seems a good choice for an analysis of RNAs which hardly undergo crystallization [129]. Tertiary structure determination procedure using NMR starts with the acquisition of multidimensional correlation spectra which are analyzed in order to determine the structure. The procedure that assigns observed NMR signals to the corresponding protons and other nuclei is a fundamental step of the RNA structure determination process. The assignment is usually based on the analysis of one and multi-dimensional spectra resulting from NMR experiments. This step is highly dependent on the experimenters knowledge, experience and intuition, and for this reason it is often not fully automated. Existing applications dedicated to RNA require an effort in data preparation to improve the quality of assignment [24]. The situation is different in case of proteins. Since studying the structures of these biomolecules is much easier, the development of methods dedicated to their exploration has been, for years, more dynamic. Automatic design of NMR spectra analysis has made a strong impact on the elucidation of protein structures [129].

At present, several softwares for protein signal assignment in two-dimensional spectra have been implemented [8, 11, 77, 98, 102, 120, 146]. Unfortunately, they appeared not suitable for processing RNA data. Hopefully, it will popularize within RNA domain soon. To the best of our knowledge, only one automatic assignment method dedicated exactly for RNA chains has been developed. The method, called RNA Probabilistic Assignment of Imino Resonance Shifts (RNA-PAIRS), predicts the secondary structure of a RNA imino resonances. RNA-PAIRS sets in motion a dynamic network that reverberates between predictions and experimental evidence in order to reconcile and rectify resonance assignments and secondary structure information. The procedure is halted when assignments and base-parings are deemed to be most consistent with observed cross-peaks [12].

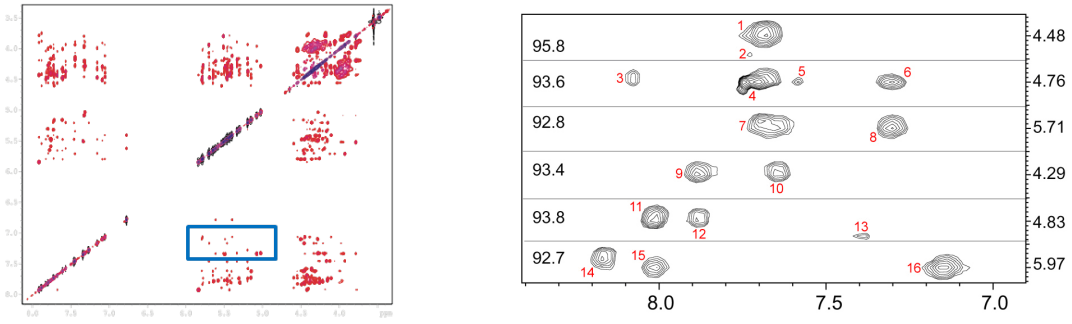
In conclusion, the development of new procedures assigning to appropriate nuclei the corresponding signals is crucial for improving the state of the art of RNA structure analysis.

3.2 The 3D assignment pathway problem

The assignment, being the first computational step in elucidation of RNA tertiary structure, is the foundation of the whole procedure. In brief, the assignment is based on the NMR spectrum, where the magnetization transfer between the nuclei along the chain should be sketched. In this section we introduce the biophysical problem of resonance assignment on 3D NMR maps, originally introduced in [129], which was an inspiration for our work, as described in [49, 127]. Finally, we discuss the computational complexity of this original problem.

3.2.1 Problem description

The Nuclear Magnetic Resonance is a technique used to study the emission of molecular electromagnetic radiations. A 3D NMR map is a record of NMR interactions that occur between RNA atoms involved in the experiment and it reflects inter-nucleic transfer of magnetization.
As previously introduced in Section 2.4.2, a 3D NMR spectroscopy experiment is a triple resonance experiments because it involves atoms (that is, the nuclei) of three different chemical elements (e.g. $H4', C4', P$) or protons (hydrogen's isotopes). In output an one-dimensional spectrum (as the one in Figure 2.2) is returned for each chemical element. All these spectra can be represented together in a unique three-dimensional spectrum whose axes are the signal frequency ranges of each 1D spectrum. The obtained 3D NMR spectrum (also called map) is an absorption spectrum containing information about resonance signals of all triplets of nuclei detected during the NMR experiment. Each correlation signals, or just *interactions*, is displayed in the spectrum by a *cross-peak*. The cross-peaks are symmetrical along the diagonal of the spectrum, and their positions represent the chemical shift values of the nuclei involved in the magnetization transfer. Any cross-peak is characterized by its location on the map (the three coordinates of its center x, y, z given in ppm), its size (i.e. width in each dimension), and a value of the relative signal intensity (the volume). Figure 3.1 represents a 3D NMR spectrum projected on the plane whose axes are the chemical shift ranges of two of the three chemical elements, and an enlarged fragment of it (the small rectangle) with enumerated cross-peaks. This NMR data can be used to compute structural parameters, like atom relative positions, inter-atomic distances, etc, which are in turn used to compute the whole three-dimensional shape of the molecule. These values are determined by coordinates of the corresponding cross-peaks. However, the relationship between cross-peaks collected on the NMR map and atoms of the analyzed molecule is not known. Therefore, the first step in the computation of the molecule structure is the assignment



(a) (b)

FIGURE 3.1: Absorption spectrum of a 3D NMR experiment projected on the plane x, y (a) and one its fragment with enumerated cross-peaks (b).

of the corresponding atoms to any cross-peaks [129]. The assignment of cross-peaks on the 3D NMR map to the corresponding atoms results from a reconstruction of particular pathway(s) between the cross-peaks. Since the atoms in the molecule are close in space, it is easy to understand that, if we could determine a path between the cross-peaks on the NMR map, then we could reconstruct the magnetization transfer track among the corresponding atoms yielded during the NMR experiment. This reconstruction allows us to labeling the cross-peaks according to the type of atoms and/or residue. Obviously, the pathway must be the longest possible with respect to the biological meaning of the problem.

Each NMR experiment induces different correlation signals among specific nuclei of the RNA chain. Therefore, the reconstruction of a sequential assignment pathway depends on the type of NMR experiment. There are three types of 3D NMR experiments:

- the homonuclear experiment (e.g. NOESY-NOESY) stimulates interactions between the nuclei of isotopes;
- the heteronuclear one (e.g. HCP) involves the nuclei of different chemical elements;
- the mixed experiment (e.g. HSQC-NOESY) combines both.

Each of these types identifies a sequence-specific connectivity pathway representing magnetization transfer between the selected nuclei of the analyzed molecule. Consequently, $H4' - C4' - P$ signals in heteronuclear HCP spectrum, representing the sequence of intra- and internucleotide[2] scalar interactions form the pathway $(H4'_n - C4'_n - P_n) - (H4'_n - C4'_n - P_{n+1}) - (H4'_{n+1} - C4'_{n+1} - P_{n+1}) - \ldots$, where n stands for a residue number [129]. An example is given in Figure 3.2, where the track of magnetization

[2] An *intranucleotide* interaction occurs when the cross-peak represents the resonance signal between adjacent nuclei that belong to the same residue; otherwise the interaction is *internucleotide*.

transfer within the single RNA chain (a) and the corresponding cross-peaks in the 3D NMR spectrum (b) are represented. Intranucleotide interactions are colored green, whereas internucleotide interactions are colored red. The reconstruction of the sequence of intra- and internucleotide interactions is the path among the cross-peaks alternated in these two colors (b). HSQC-NOESY is mixed, homo- and heteronuclear experi-

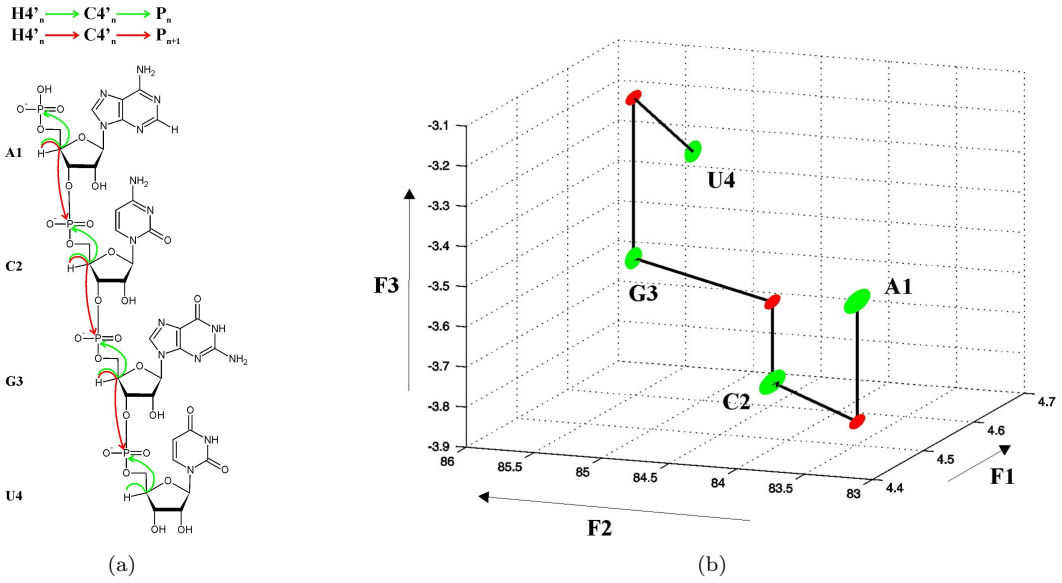


FIGURE 3.2: A fragment of simulated 3D HCP spectrum for r(ACGU) with the magnetization transfer pathway between $H4' - C4' - P$ nuclei. F1,F2,and F3 axes in (b) represent chemical shift ranges ranges of $H4' - C4' - P$ respectively.

ment, being most frequently used to resonance assignment of RNAs. It provides the information about many different interactions, collected in the separate regions of its spectrum. The most meaningful are the signals constructing the following pathways: $(C1'_n - H1'_n - H8/H6_n) - (C1'_n - H1'_n - H8/H6_{n+1}) - (C1'_{n+1} - H1'_{n+1} - H8/H6_{n+1}) - \ldots$, and $(C8/C6_n - H8/H6_n - H1'_n) - (C8/C6_n - H8/H6_n - H1'_{n+1}) - (C8/C6_{n+1} - H8/H6_{n+1} - H1'_{n+1}) - \ldots$ [129]. Finally, homonuclear NOESY-NOESY spectra can be used for a reconstruction of several magnetization transfer tracks, from which crucial are $(H8/H6_{n+1} - H1'_n - H8/H6_n) - (H8/H6_{n+1} - H1'n - H8/H6_{n+1}) - (H8/H6_{n+1} - H1'_{n+1} - H8/H6_{n+1}) - \ldots$ [129]. Consequently, two types of NMR interactions can be observed: homonuclear (in homonuclear and mixed experiments) and heteronuclear (in heteronuclear and mixed experiments). Similarly, there are two types of assignment pathways.

Pathway construction starts from any cross-peak $p^i(x^i, y^i, z^i)$ on the map, where x^i, y^i, z^i are the coordinates of p^i. Next, the following steps are repeated depending on the pathway type:

a. for a heteronuclear pathway

(1) go to p^{i+1} if $x^{i+1} = x^i$, $y^{i+1} \neq y^i$ and $z^{i+1} \neq z^i$

(2) go to p^{i+2} if $x^{i+2} \neq x^{i+1}$, $y^{i+2} = y^{i+1}$ and $z^{i+2} = z^{i+1}$.

Other combinations are also possible in the heteronuclear case, provided that in the first step any two coordinates of p^{i+1} must equal their counterparts in p^i, while in the second step - the remaining one must satisfy the equality.

b. for a homonuclear pathway

(1) go to p^{i+1} if $x^{i+1} = x^i$, $y^{i+1} = y^i$ and $z^{i+1} \neq z^i$

(2) go to p^{i+2} if $x^{i+2} \neq x^{i+1}$, $y^{i+2} = y^{i+1}$ and $z^{i+2} = z^{i+1}$

(3) go to p^{i+3} if $x^{i+3} = x^{i+2}$, $y^{i+3} \neq y^{i+2}$ and $z^{i+3} = z^{i+2}$.

The above steps are iterated as long as there are unvisited cross-peaks on the NMR map and the pathway can be extended (i.e. there is at least one unvisited cross-peak that has the appropriate coordinates and so it can be added to the pathway). The objective is to find the longest path under the constraint that each cross-peak can be visited at most once.

3.2.2 Problem computational complexity

In this section we discuss the computational complexity of the assignment pathway construction on the 3D NMR map. The problem is proved NP-hard. Below we report our proof presented in [127].

Let us first consider the heteronuclear case. We define a decision version of the problem of finding the 3D heteronuclear assignment pathway (denoted by Π') in the following way:

Instance:
A 3D NMR graph, i.e. graph $G' = (V', E')$ located in the three-dimensional space, where $V' = \{\omega^i(x^i, y^i, z^i) : i = 1, \ldots, n\}$ represents a set of n cross-peaks from the corresponding 3D NMR map, each vertex has three coordinates x, y, z (equal to those of the relative cross-peak), E' is a set of edges (an edge is built between every two vertices, $\omega^i, \omega^j \in V'$, which have either one or two common coordinates).

Question:
Does G' contain a heteronuclear assignment pathway, that is an ordering $P = \langle \omega^1, \omega^2, \ldots, \omega^n \rangle$ of vertices in G', such that every vertex occurs in P at most once, $(\omega^i, \omega^{i+1}) \in E'$ for

all i $(1 \leq i < n)$, and any three-element subsequence $\langle \omega^i, \omega^{i+1}, \omega^{i+2} \rangle \in P$ satisfies the rules: $x^i = x^{i+1}, x^{i+1} \neq x^{i+2}, y^i \neq y^{i+1}, y^{i+1} = y^{i+2}, z^i \neq z^{i+1}, z^{i+1} = z^{i+2}$?

Proof. In order to prove that $\Pi' \in NP$, it is sufficient to demonstrate a nondeterministic algorithm solving the problem in polynomial time. Such algorithm only needs to guess an ordering of vertices in G' and check in polynomial time whether all the relationships between vertex coordinates are satisfied.
Next, let us take the known NP-complete problem Π of finding a Hamiltonian path in a given graph $G = (V, E)$ that will be used to transform Π' into Π. In the decision version of Hamiltonian path problem the question is whether graph G contains an ordering $\langle v_1, v_2, \ldots, v_k \rangle$ of its vertices, such that $k = |V|$ and $(v_i, v_{i+1}) \in E$ for all i $(1 \leq i < k)$. The problem remains NP-complete even if we assume that G has no self-loops and no vertices with degree exceeding three [65]. Now, taking an arbitrary graph $G = (V, E)$, that is an instance of the Hamiltonian path problem, we can construct G' in the following way (cf. Figure 3.3):

(1) For every vertex $v_i \in V$ in G, place two corresponding vertices in G': $\omega_0^i(x_v, y_v, 0) \in V'$ and $\omega_1^i(x_v, y_v, 1) \in V'$.

(2) For every edge $e_j(v_p, v_t) \in E$ in G, place two corresponding edges in G': $e_0^j = (\omega_0^p, \omega_0^t)$ and $e_1^j = (\omega_1^p, \omega_1^t)$.

(3) For every vertex $v_i \in V$ in G, introduce edge $e_{01}^i = (\omega_0^i, \omega_1^i)$ in G'.

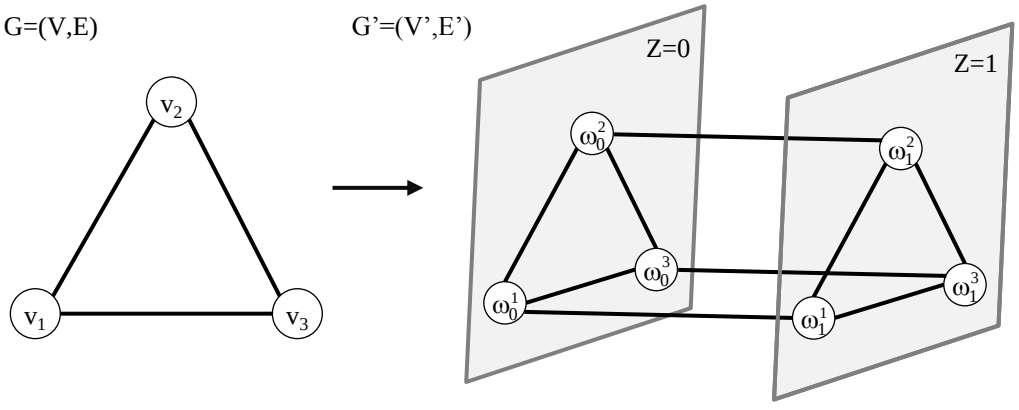


FIGURE 3.3: Construction of 3D NMR graph (heteronuclear case).

By that means, we obtain 3D NMR graph $G' = (V', E')$, where $V' = V_0' \cup V_1'$, $|V'| = 2|V|$, and $E' = E_0' \cup E_1' \cup E_{01}'$, $|E'| = 2|E| + |V|$. The time used to construct G' is bounded from the above by the input length of problem Π. Based on the above transformation we can

observe that G contains a Hamiltonian path if and only if the corresponding 3D NMR graph G' contains a heteronuclear assignment path. This in turn implies NP-hardness of the heteronuclear assignment pathway construction on the 3D NMR map. □

Similarly we can treat the problem of finding a 3D homonuclear assignment path. The 3D NMR graph $G'' = (V'', E'')$ to represent this version of the problem is constructed as in the heteronuclear case, except that every edge $(\omega^i, \omega^j) \in E''$ connects two vertices from V'', which have exactly two common coordinates. In the decision version (denoted by Π''), the question is formulated as follows:

Question:
does G'' contain a homonuclear assignment pathway, that is an ordering $P = \langle \omega^1, \omega^2, \ldots, \omega^n \rangle$ of vertices in G'', such that every vertex occurs in P at most once, $(\omega^i, \omega^{i+1}) \in E$" for all i $(1 \leq i < n)$, and any four-element subsequence $\langle \omega^i, \omega^{i+1}, \omega^{i+2}, \omega^{i+3} \rangle \in P$ satisfies the rules: $x^i = x^{i+1}, x^{i+1} \neq x^{i+2}, x^{i+2} = x^{i+3}, y^i = y^{i+1}, y^{i+1} = y^{i+2}, y^{i+2} \neq y^{i+3}, z^i \neq z^{i+1}, z^{i+1} = z^{i+2}, z^{i+2} = z^{i+3}$?

Proof. The proof is again based on the transformation using the Hamiltonian path problem Π. Given $G = (V, E)$, being an instance of Π, we construct $G'' = (V'', E'')$ according to the following steps:

(1) For every vertex $v_i \in V$ in G, place the corresponding vertex in G'': $\omega^i(x_v, y_v, z_v) \in V''$.

(2) For every edge $e_j(v_p, v_t) \in E$ in G, construct a cubic subgraph S''_j (cf. Figure 3.4) and add it to G'' between vertices $\omega^p, \omega^t \in V''$.

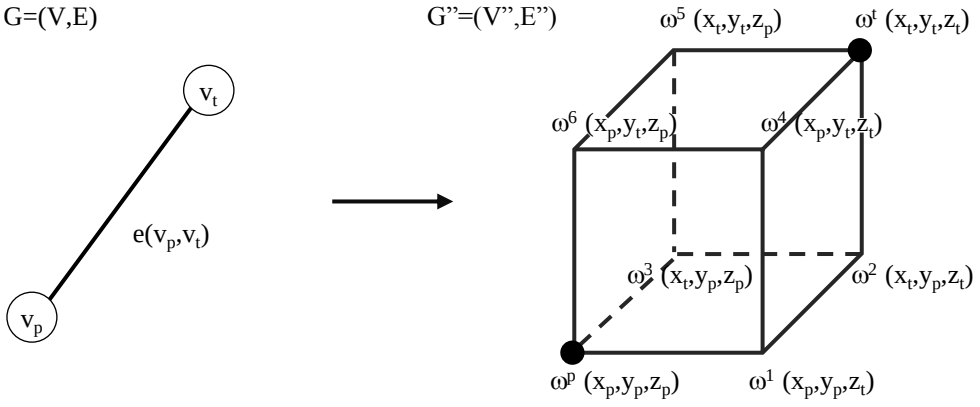


FIGURE 3.4: Subgraph construction in 3D NMR graph (homonuclear case).

Again, the time used to construct G'' is bounded from above by the input length of problem Π. Moreover, from the transformation it is clear that G contains a Hamiltonian

path if the corresponding graph G'' contains a homonuclear assignment path, and vice versa. This proves the NP-hardness of the homonuclear assignment pathway problem. □

3.3 An edge-colored graph based formulation

Several problems modeled by edge-colored graphs have achieved significant interest during last decades, both from the theoretical point of view and of its domains of applications. Although many problems may be modelled by seeking multicolored cycles (i.e., cycles with edges of different color) or monochromatic cycles (i.e., cycles whose edges have the same color) [34, 93], usually the most hype field of study concerns the minimization of the number of colors on the graph.

According to the instance provided as input, we can distinguish between problems defined on edge-colored graph and problems whose aim is to define an optimal coloring for a given uncolored graph. The latter are called *edge-coloring problems*, and a feasible solution of them is an assignment of colors to the edges of the graph. Usually, the assignment is assumed to be a proper coloring of the edges (i.e., no two adjacent edges have assigned the same color), which is obtained using the least number of colors. Edge coloring problems have applications in scheduling problems [64, 139] and in fiber optic networks [81], among others. For example, a communication network may be represented as a graph where any vertex is a station and an edge connects two stations that wish to communicate with each other by a fiber optic. Different frequency of light are indicated with different colors. The problem of assigning colors to the edges in such a way that no two paths that share a segment of fiber optic use the same frequency is a *path coloring problem*, which was first introduced by Aggarwal et al. [2] and Raghavan and Upfal [114]. A particular path coloring problem is the *non-repetitive coloring* problem [5], where a coloring is a sequence of colors on any path in the graph such that a path cannot contain adjacent subpaths colored with the same sequence of colors.

On the other hand, when a coloring for a given graph G is already known, one may be interested in extracting subgraphs colored in a specified pattern. For example, a transportation network can be represented by a graph whose edges are colored according to the different modes of transportation, then to seek a path of minimum number of colors between two vertices is equal to look for a path connecting them by using the minimum number of different modes of transportation [40]. This is a typical application of the well-known minimum label spanning tree problem, initially addressed by Broersma and Li [33] and Chang and Leu [41]. This problem falls in the class of the combinatorial optimization problems defined on labeled graphs: the *label graph problems* (see [36, 39, 59, 143], to name a few).

In this section we describe the OCLP problem based on edge-colored graph representing the assignment pathway problem presented in Section 3.2. As previously described in [127], we build an edge-colored graph $G = (V, E)$ from a 3D NMR spectrum such that any vertex $v \in E$ is a cross-peak, and the edges are colored according to the type of interaction occurring in the NMR experiment. Finally, we consider other possible applications of the OCLP problem at the outside of the biological world. However, this last part is under completion in [128].

3.3.1 Notation and Definitions

Definition 3.1. Let $G = (V, E)$ be a finite undirected simple graph, and $C = \{1, \ldots, c\}$ be a set of colors, with $c \geq 2$. G is a *c-edge-colored* graph if any edge $\{v_i, v_j\} \in E$ is colored from C.

Definition 3.2. Let $G = (V, E)$ be a c-edge-colored graph. A path in G where any two adjacent edges differ in color is an *properly colored* path.

Definition 3.3. Let $G = (V, E)$ be a c-edge-colored graph. An *orderly colored* path (OCP) in G is a properly colored path whose edges follow a given sequence of colors.

Definition 3.4. Let $G = (V, E)$ be a c-edge-colored graph. An *alternating* path in G is a orderly colored path whose edges are alternated in a sequence of only two colors.

The Orderly Colored Longest Path Problem
Let $G = (V, E)$ be a c-edge-colored graph on a set of colors $C = \{1, \ldots, c\}$. Given a subset $Q \subseteq C$ of k colors, where $2 \leq k \leq c$, and an order on Q. Let $S \subseteq E$ be the subset containing the edges colored with the colors in Q. The orderly colored longest path problem is the problem of finding the path of maximum length in $G[S]$ such that edge colors along the path hold the order on Q.

The Orderly Colored Hamiltonian Path Problem
The orderly colored hamiltonian path problem is the problem of finding an OCLP which visits each vertex in G exactly once.

Let us consider the small graph in Figure 3.5.b. composed of 6 vertices, 10 edges, and 3 colors, where $Q \subseteq C = \{green, blue, red\}$, and the ordering on Q is the sequence given in Figure 3.5.a. The longest orderly colored path (OCLP) is the orderly colored hamiltonian path (OCHP) represented in Figure 3.5.c. For simplicity, we assume that a feasible path must always start from a given color (green, in this case); such assumption can be easily removed by solving c instances of the same model.

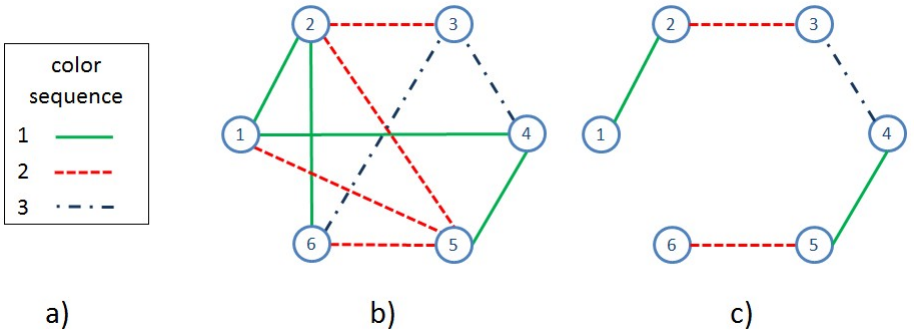


FIGURE 3.5: A simple example of 3-edge-colored graph. For better readability, green arcs are solid, red arcs are dotted, blue arcs are dashes.

3.3.2 Building the edge-colored graph

With respect to the description of Section 3.2.1, the problem can be modeled with an edge-colored undirected graph $G = (V, E)$. Every vertex $v^i \in V$ represents cross-peak p^i from the 3D NMR map and $|V| = n$, where n equals the number of cross-peaks. Every connection between two cross-peaks, which have either one or two common coordinates, is represented by an edge $e \in E$ in graph G. Thus, if m is the number of all possible connections that can be traced on NMR map, then $|E| = m$. Each edge $e(v^i, v^j) \in E$ is assigned a color (label) from the set of $c = 6$ colors $C = \{0, 1, 2, 3, 4, 5\}$, and each color represents different relationship between the coordinates of the corresponding cross-peaks. Thus, E can be partitioned into six subsets:

$$
\begin{aligned}
E_0 &= \{(v^i, v^j) \in E : \quad x^i \neq x^j,\ y^i = y^j,\ z^i = z^j\} \\
E_1 &= \{(v^i, v^j) \in E : \quad x^i = x^j,\ y^i \neq y^j,\ z^i = z^j\} \\
E_2 &= \{(v^i, v^j) \in E : \quad x^i = x^j,\ y^i = y^j,\ z^i \neq z^j\} \\
E_3 &= \{(v^i, v^j) \in E : \quad x^i = x^j,\ y^i \neq y^j,\ z^i \neq z^j\} \\
E_4 &= \{(v^i, v^j) \in E : \quad x^i \neq x^j,\ y^i \neq y^j,\ z^i = z^j\} \\
E_5 &= \{(v^i, v^j) \in E : \quad x^i \neq x^j,\ y^i = y^j,\ z^i \neq z^j\}
\end{aligned}
$$

Such c-edge-colored graph G represents the 3D NMR map together with all passages that can be crossed when the assignment pathway is constructed (see Figure 3.6). Keeping in mind that we consider three types of 3D NMR maps (homonuclear, heteronuclear and mixed), we understand how they are represented by different c-edge-colored graphs:

a. If $G = (V, E)$ encodes a heteronuclear or a mixed map: $c = 6$, $E = E_0 \cup E_1 \cup E_2 \cup E_3 \cup E_4 \cup E_5$.

b. If $G = (V, E)$ encodes a homonuclear map: $c = 3$, $E = E_0 \cup E_1 \cup E_2$.

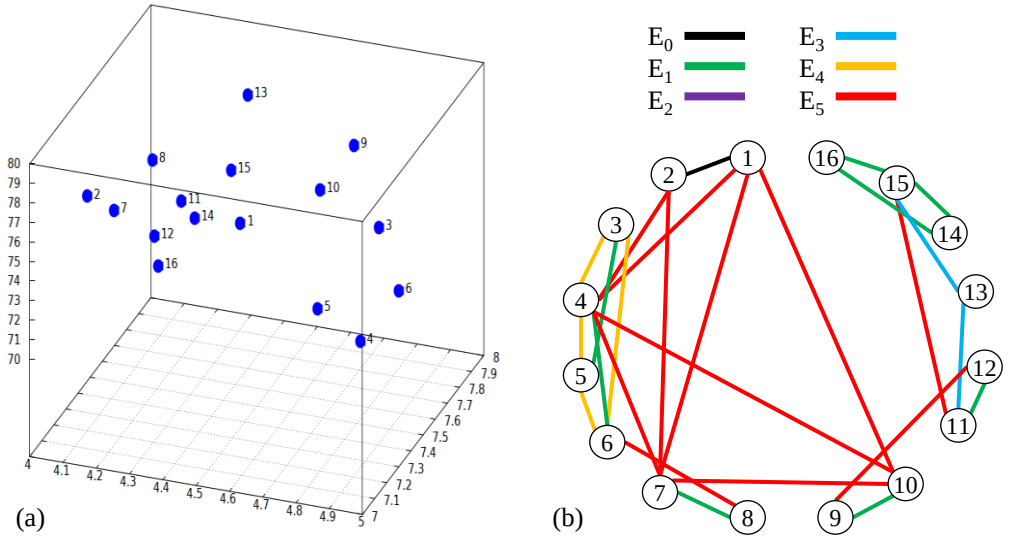


FIGURE 3.6: Example 3D NMR map (a) and its representation as edge-colored graph (b). x, y, z axes in (a) represent chemical shift ranges of each nucleus.

Due to the description in Section 3.2.1 two types of assignment pathways (homonuclear or heteronuclear) can be reconstructed in c-edge-colored graph representing the 3D NMR map. Thus, we define two versions of 3D-APP.

3D-APP$_{HE}$ (heteronuclear assignment pathway problem):
Given a c-edge-colored undirected graph $G = (V, E)$, where V is a set of vertices representing cross-peaks from the heteronuclear or mixed 3D NMR map, and E is a set of colored edges partitioned into c=6 subsets, find the longest elementary path $P = \{e_0, e_1, \ldots, e_k\}$ in E such that if $e_i \in E_w$ then $e_{i+1} \notin E_w$, and the edges of P are alternately in either E_0, E_3 or E_1, E_5 or E_2, E_4.

3D-APP$_{HO}$ (homonuclear assignment pathway problem):
Given a c-edge-colored undirected graph $G = (V, E)$, where V is a set of vertices each of which represents a cross-peak from the homonuclear 3D NMR map, and E is a set of colored edges partitioned into c subsets ($c \in \{3, 6\}$), find the longest elementary path $P = \{e_0, e_1, \ldots, e_k\}$ in $E_0 \cup E_1 \cup E_2$ such that if $e_i \in E_w$ then $e_{i+1}, e_{i+2} \notin E_w$, and the edges of P are alternately in E_0, E_1, E_2.

In conclusion, the problem of finding the longest transfer pathway between the cross-peaks in order to reconstruct the sequential assignment of NMR signals on the 3D NMR map can be represented as the OCLP problem, where the path is orderly in a sequence of either 2 (2-OCLP) or 3 (3-OCLP) colors, respectively.

2-OCLP as reformulation of the 3D-APP$_{HE}$:
Let $G = (V, E)$ be a 6-edge-colored graph on a set of colors $C = \{1, \ldots, 6\}$, where V is a

set of vertices representing cross-peaks from the heteronuclear or mixed 3D NMR map. Given a subset $Q \subseteq C$ of 2 colors representing the only possible steps of the assignment pathway. Let $S \subseteq E$ be the subset containing the edges colored with the colors in Q. Find the longest alternating path in $G[S]$.

3-OCLP as reformulation of the 3D-APP$_{HO}$:
Let $G = (V, E)$ be a 6-edge-colored graph on a set of colors $C = \{1, \ldots, 6\}$, where V is a set of vertices representing cross-peaks from the homoronuclear 3D NMR map. Given a subset $Q \subseteq C$ of 3 colors representing the only possible steps of the assignment pathway, and an order on Q. Let $S \subseteq E$ be the subset containing the edges colored with the colors in Q. Find the longest path in $G[S]$ such that the order on Q is hold.

3.3.3 Not only biological applications

Recently a few applications of properly edge-colored Hamiltonian and Eulerian cycles and paths in molecular biology have been studied [54, 55, 112, 113]. In particular, in [55] is discussed the existence of alternating Hamiltonian circuits that determine the spatial order of the chromosomes within haploid chromosome complements; whereas in [113] alternating Eulerian cycles are used to represent the solutions of the Double Digest Problem (i.e., the problem of constructing a physical maps of DNA sequences). Nevertheless, we believe that besides the 3D-APP, other well-known problems could be formulated as an OCLP problem both in biology and in other fields.
For example, the OCLP problem may be used for modeling city blocks problems on *mesh graphs* where intersections are vertices and streets are edges [108]. Let us consider a rectangular grid where horizontal edges are colored with one color and vertical edges with the other one, then a path whose edges are alternated in dimension will also be a path whose edges are alternated in color. Analogously, a mesh structure may represent a chessboard setting a vertex on each square and coloring the edges from a set of colors. Hence we can model edge-colors constrained path problems related to the chess moves, such as for the knight's tour problem [136], which is the problem of seeking an Hamiltonian (or longest) path composed of knight's moves. Of interest are also the applications on grid graphs that contain obstacles, e.g., forbidden vertices [130], such as in the Longest Path Routing problem discussed in [131]. Similarly, orderly colored schemes could be useful to model particular routing problems: pick-up and delivery, where different types of pick-up and deliveries must alternate over the edges [117]; and electric vehicles routing, where loading and discharging arcs must alternate, or separated waste collection routing [116].

3.4 Three IP formulations

In the previous section we have shown how the 3D-APP can be formulated as OCLP problem. Here, we describe three integer programming (IP) formulations for modeling the OCLP problem such as the network problems presented in [49]. The edge-colored graph G is transformed by coping k times the vertex set V in order to obtain a k-partite digraph, on which we impose additional packing constraints.
The first model is based on a longest path problem in a n-partite graph with a number of partitions equal to the number of vertices of the original graph. The second model is a transformation of the latter, where the number of partite sets depends on the number of colors. The third model is formulated on a n-partite graph, where each partition is a c-connected subgraph of the original graph. In the following we assume, without loss of generality, that the searched path must always start from color with label 1.

3.4.1 Longest path over acyclic n-partite graph

Let $G = (V, E)$ be a c-edge-colored graph. We transform G to digraph $D = (V', A)$ as follows:

1. The set of vertices $V = \{v^1, v^2, \ldots, v^n\}$ from G is repeated n times in V' such that (i) $V' = V^1 \cup V^2 \cup \cdots \cup V^n$ and $V^i \cap V^j = \emptyset$, for each $i, j \in V : i \neq j$; and (ii) $V^l = \{v_1^l, v_2^l, \ldots, v_n^l\}$ for each $l \in \{1, \ldots, n\}$. We refer to each V^l as a *partite set*.

2. We denote a *level set* as $L_r = \{v_r^1, v_r^2, \ldots, v_r^n\}$ such that any vertex v_r^l belongs to a different partite set, thus $V^l \cap L_r = \{v_r^l\}$ for each $r = 1, \ldots, n$ and $l \in \{1, \ldots, n\}$.

3. Let (L_r, L_{r+1}) be a pair of two consecutive level sets. Each edge $(v^i, v^j) \in E$ is replaced by two arcs in A, (v_r^i, v_{r+1}^j) and (v_r^j, v_{r+1}^i), in order to connect (L_r, L_{r+1}). Both arcs have the same cost c_{ij} associated with $(v^i, v^j) \in E$.

4. Each pair of level sets (L_r, L_{r+1}) is connected by arcs of the same color from G.

   On the basis of the color sequence requested from the path, in the $n-1$ level set pairs the arcs are recursively alternated in color. E.g., if $c = 2$ then all pairs of arcs $(v_1^i, v_2^j) \in A$ and $(v_1^j, v_2^i) \in A$ in (L_1, L_2) are from those edges $(v^i, v^j) \in E$ with color 1 in G; all pairs of arcs $(v_2^i, v_3^j) \in A$ and $(v_2^j, v_3^i) \in A$ in (L_2, L_3) are from those edges $(v^i, v^j) \in E$ with color 2 in G; all pairs of arcs $(v_3^i, v_4^j) \in A$ and $(v_3^j, v_4^i) \in A$ in (L_3, L_4) are from those edges $(v^i, v^j) \in E$ with color 1 in G and so on.

5. One source vertex s and one destination vertex t are added to D, thus $|V'| = n^2+2$; $L_0 = \{s\}$; $L_{n+1} = \{t\}$. Arcs with null weight connect s with each vertex in L_1, and each vertex in V' to t (the latter referred to as *exit arcs*).

Summarizing, any vertex $v^l \in V$ has n copies in V^l, each of which is in turn an element of a level set L_r.

Example 3.1. *Let us consider the* 3*-edge-colored graph* $G = (V, E)$ *in Figure 3.5. The corresponding network* D *has* 38 *vertices and* 6 *partite sets as shown in Figure 3.7. The grey horizontal box represents the first partite set containing* 6 *copies of the vertex* 1*, whereas the grey vertical box represents the fifth level set containing one copy for each vertex of* V*. The arcs in each level set pair are alternated w.r.t. the ordering of colors* {green, red, blue}*. We remark that the arcs represented by light grey lines are the exit arcs; for better readability in Figure 3.7 only exit arcs from vertices belonging to the first and the last partite sets have been represented; although the model accounts for exit arcs from each one of the 36 vertices to the sink vertex.*

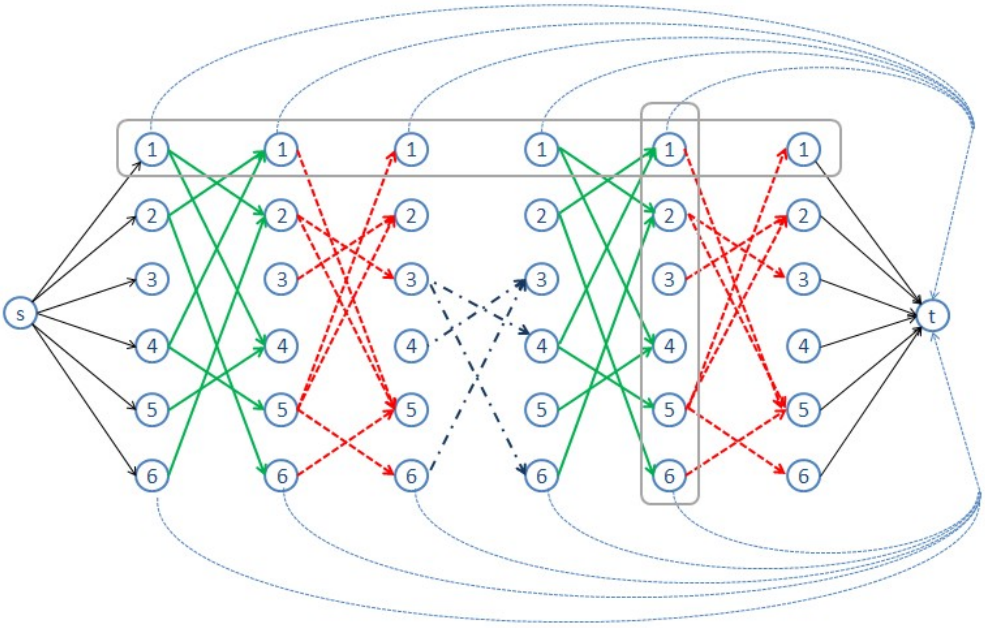


FIGURE 3.7: The 6-partite graph arising from the edge-colored graph of Figure 3.5.

Theorem 3.1. *Let* $D = (V', A)$ *be a* n*-partite digraph as above defined. If* D *admits a hamiltonian path from* s *to* t *such that successive vertices belong to different partite sets* V^l*, then this path is an OCHP in* G*.*

Proof. Let us assume that for each partite set V^l in D we have $|V^l| = 1$. Then a hamiltonian path in from s to t in D is also a hamiltonian path in G. By definition

we know that any arc (v^i, v^j) connecting successive level sets (L_r, L_{r+1}) is colored in G according to the given sequence of colors composing the path, and since $|V^l| = 1$ then $|L_r| = |V^l|$. Therefore, the hamiltonian path in D such that each arc is $(v^i, v^j) : v^i \in V^l, v^j \in V^{l+1}$ is orderly colored in G. □

Corollary 3.1. *An OCHP in G is an OCLP in G composed by $n-1$ edges.*

Theorem 3.2. *Let $D = (V', A)$ be a n-partite digraph as above defined. Then a longest path from s to t in D such that successive vertices belong to different partite sets V^l is an OCLP in G.*

Proof. By Corollary 3.1 instances of the OCLP problem form a subclass of instances of the OCHP problem. □

We now describe the corresponding IP formulation. Let $D = (V', A)$ be a directed graph as above described. For each arc $(v_r^i, v_{r+1}^j) \in A$ we associate the decision variable x_r^{ij}, where $x_r^{ij} = 1$ if the arc (v_r^i, v_{r+1}^j) belongs to the path P, and $x_r^{ij} = 0$ otherwise. All arc costs are set $c_r^{ij} = 1$ except for those incident to s or t, which is assigned zero cost. The subscript r is used to distinguish the same arcs repeated in different level set pairs.
The longest path over acyclic n-partite graph (LPnPP) is an optimization model formulated as follows:

Maximize $\sum_{(v_r^i, v_{r+1}^j) \in A} c_r^{ij} x_r^{ij}$

subject to:

$$\sum_{(v_r^i, v_{r+1}^j) \in A} x_r^{ij} - \sum_{(v_{r-1}^j, v_r^i) \in A} x_r^{ji} = 0 \quad \forall v_r^i \in V' - \{s, t\} \quad (C1)$$

$$\sum_{(s, v_1^j) \in A} x_1^{sj} = 1 \quad (C2)$$

$$\sum_{(v_r^i, t) \in A} x_r^{it} = 1 \quad (C3)$$

$$\sum_{\substack{(v_{r-1}^j, v_r^i) \in A \\ v_r^i \in V^l}} x_r^{ji} \leq 1 \quad l = 1, 2, \ldots, n \quad (C4)$$

$$x_r^{ij} \in \{0, 1\} \quad \forall (v_r^i, v_{r+1}^j) \in A \quad (C5)$$

where the constraints (C1-C3) are the classical balance constraints formulated on a network problem: from the source s to the sink t, anytime a vertex v_r^i is reached from the path P the next vertex selected v_{r+1}^j must be adjacent to v_r^i. The set of *packing constraints* (C4) ensure that for each set V^l at most one vertex can be visited by the path. In this way only one copy of the same vertex of the original graph is visited by the path, making us sure that the expansion of the vertices does not create unfeasible paths w.r.t. the original problem. According to these consideration, it is easy to state the following:

Preposition 3.1. *A feasible (optimal) solution x^* of LPnPP in D is an orderly colored (longest) path in G. If x^* is composed by $n-1$ edges, then it is an OCHP in G.*

3.4.2 Longest path over cyclic c-partite graph

In this second formulation we significantly simplify the dimension of the graph used to search the longest paths introducing cycle elimination constraints in the formulation. These constraints can be separated, making this approach potentially interesting from the computational point of view, as will be detailed in Section 3.5.1.

Let $G = (V, E)$ be a c-edge-colored graph. We transform G to digraph $D = (V', A)$ as follows:

1. The set of vertices $V = \{v^1, v^2, \ldots, v^n\}$ from G is repeated c times in V' such that (i) $V' = V^1 \cup V^2 \cup \cdots \cup V^n$ and $V^i \cap V^j = \emptyset$, for each $i, j \in V : i \neq j$; and (ii) $V^l = \{v^l_1, v^l_2, \ldots, v^l_c\}$ for each $l \in \{1, \ldots, n\}$. We refer to each V^l as a *partite set*.

2. We denote a *level set* as $L_r = \{v^1_r, v^2_r, \ldots, v^n_r\}$ such that any vertex v^l_r belongs to a different partite set, thus $V^l \cap L_r = \{v^l_r\}$ for each $r = 1, \ldots, c$ and $l \in \{1, \ldots, n\}$.

3. Let (L_r, L_{r+1}) be a pair of two consecutive level sets. Arcs are directed from L_r to L_{r+1}, $r = 1, \ldots, c$. Each edge $(v^i, v^j) \in E$ is replaced by two arcs in D, (v^i_r, v^j_{r+1}) and (v^j_r, v^i_{r+1}), in order to connect the consecutive level sets (L_r, L_{r+1}). Both arcs have the same cost c_{ij} from the edge $(v^i, v^j) \in E$.

   Differently from LPnPP, here the number of level set pairs is $c-1$, instead of $n-1$. Indeed, the last pair of level sets is composed by (L_c, L_1), because the arcs labeled with color c connecting the last level set L_c to the first one L_1. Therefore D' may be considered partitioned respect the number of colors. Below we will indicate a level set pair with (L_p, L_q) and its arc with (v^i_p, v^j_q), where $q = rest + 1$ and $rest$ is the remainder of p/c.

4. Each pair of level sets (L_p, L_q) contains arcs with same color in G.

   On the basis of the color sequence requested from the path, in the c level set pairs the arcs are recursively alternated in color. E.g., if $c = 3$ then all pairs of arcs $(v^i_1, v^j_2) \in A$ and $(v^j_1, v^i_2) \in A$ in (L_1, L_2) are from those edges $(v^i, v^j) \in E$ with color 1 in G; all pairs of arcs $(v^i_2, v^j_3) \in A$ and $(v^j_2, v^i_3) \in A$ in (L_2, L_3) are from those edges $(v^i, v^j) \in E$ with color 2 in G and all pairs of arcs $(v^i_3, v^j_1) \in A$ and $(v^j_3, v^i_1) \in A$ in (L_3, L_1) are from those edges $(v^i, v^j) \in E$ with color 3 in G.

5. One source vertex s and one destination vertex t are added to D, thus $|V'| = nc+2$; Arcs with null weight connect s with each vertex in L_1, each vertex in V' to t (the latter referred to as *exit arcs*).

Example 3.2. *As in the example 3.1 we consider the edge-colored graph of Figure 3.5. Here, the corresponding network has* 20 *vertices,* 6 *partite sets* V^l*, only* 3 *level sets* L_1*,* L_2*, and* L_3*, as shown in Figure 3.8. The arcs between the three level sets are directed from* L_1 *to* L_2 *if their color label is* green *in* G*; are directed from* L_2 *to* L_3 *if their color label is* red *in* G*; and are directed from* L_3 *to* L_1 *if their color label is* blue *in* G*. Also here, arcs represented by light grey lines are the exit arcs, and only a part of them is represented for readability (exit arcs should be present from each vertex to the sink vertex).*

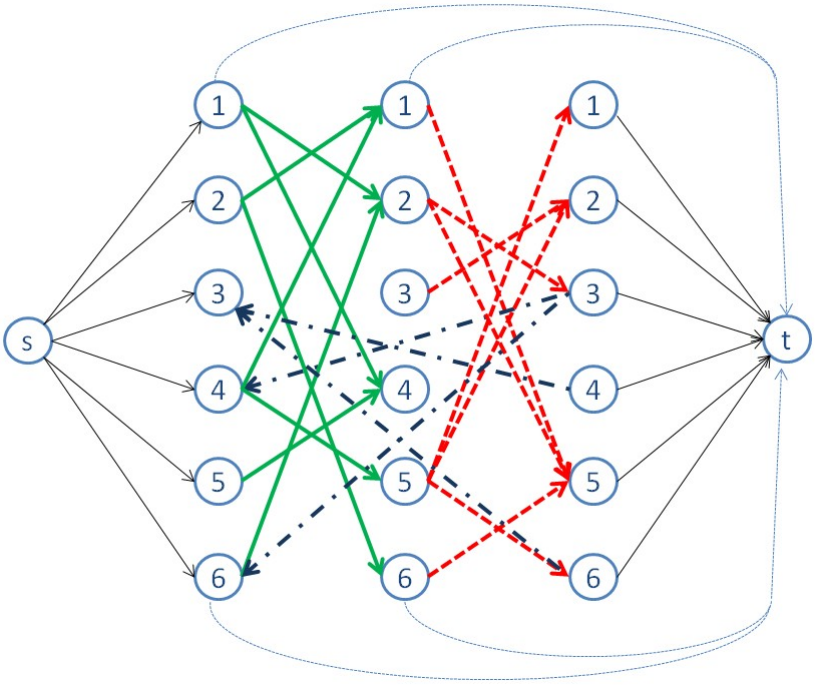


FIGURE 3.8: The 3-partite graph arising from the edge-colored graph in Figure 3.5.

Let $D = (V', A)$ be a directed graph as above described. Since in this model there are not arcs repetitions, for each arc $(v_p^i, v_q^j) \in A$ we refer with x^{ij} to the binary decision variables, and with c_r^{ij} to the arc cost. All arc costs are set $c_r^{ij} = 1$ except for those incident to s or t, which is assigned zero cost. The longest path over cyclic c-partite

graph (LPcPP) is formulated as follows:

$$
\begin{aligned}
&\text{Maximize} && \textstyle\sum_{(v_p^i,v_q^j)\in A} c^{ij}x^{ij} && && \\
&\text{subject to:} && && && \\
& && \textstyle\sum_{(v_p^i,v_q^j)\in A} x^{ij} - \sum_{(v_q^j,v_p^i)\in A} x^{ji} = 0 && \forall v_p^i \in V' - \{s,t\} && (C1)\\
& && \textstyle\sum_{(s,v_1^j)\in A} x^{sj} = 1 && && (C2)\\
& && \textstyle\sum_{(v_p^i,t)\in A} x^{it} = 1 && && (C3)\\
& && \textstyle\sum_{\substack{(v_p^i,v_q^j)\in A \\ v_p^i\in V^l}} x^{ij} \leq 1 && l = 1,2,\ldots,n && (C4)\\
& && \textstyle\sum_{(v_p^i,v_q^j)\in \Gamma} x^{ij} \leq |\,\Gamma\,| - 1 && \Gamma \in \widehat{\Gamma} && (C5)\\
& && x^{ij} \in \{0,1\} && \forall\,(v_p^i, v_q^j) \in A && (C6)
\end{aligned}
$$

As in LPnPP, the constraints (C1-C3) ensure that the solution describes a path from s to t. The packing constraints (C4) state that in any set V^l at most one vertex can be visited from the path. Since this time the network is not acyclic, we need to enforce the separations of all the orderly colored cycles; in the formulation this is achieved with constraints (C5), that for each cycle $\Gamma \in \widehat{\Gamma}$ expresses cycle elimination constraints.
Cycle elimination constraints are not added in the initial formulation of the problem when the problem is solved, but they are iteratively separated if a cycle appears in the current solution.

Preposition 3.2. *A feasible (optimal) solution x^* of LPcPP in D is an orderly colored (longest) path in G. If x^* is composed by $n-1$ edges, then it is an OCHP in G.*

3.4.3 Longest path over cyclic *c*-connected graph

Although the formulation of this third model requires cycle elimination constraints as the previous one, and the number of vertices still depends on the number of colors, here the graph is not really c-partite. For the sake of clarity, let us change the notation adopted in the first two models.
Let $G = (V, E)$ be a c-edge-colored graph. We transform G to digraph $D = (V', A)$ as follows:

1. The set of vertices $V = \{v^1, v^2, \ldots, v^n\}$ from G is repeated c times in V' such that (i) $V' = V^1 \cup V^2 \cup \cdots \cup V^n$ and $V^i \cap V^j = \emptyset$, for each $i, j \in V : i \neq j$; (ii) $V^i = \{v_1^i, v_2^i, \ldots, v_c^i\}$ for each $i \in \{1, \ldots, n\}$, and (iii) any copy $v_i^l \in V^i$ has a different color r from the set of colors $C = \{1, \ldots, c\}$.

2. The arcs set is composed by two subsets $A = A' \cup A''$, where:

- A' is in turn composed by n disjoint subsets A^i such that $A' = A^1 \cup A^2 \cup \cdots \cup A^n$, and where $|A^i| = c$ for each $i \in \{1, \ldots, n\}$.

  Arcs in subset A^i define a cycle on the V^i vertices by connecting a vertex to another in V^i according to the requested ordering of colors. When we consider the subgraph $D^i = (V^i, A^i)$ composed by the subsets of vertices V^i and the subset of arcs A^i, then we refer to a *color connected component*.

  In practice, for any pair of vertices v_l^i and v_r^i in V^i there is in A^i an arc directed from v_l^i to v_r^i, and the last arc is precisely (v_c^i, v_1^i) which closes a cycle in A^i. The sequence of the arcs depends on the requested color sequence; i.e., if the path crosses the arc $(v_l^i, v_r^i) \in A^i$ in D, then the corresponding path in G is changing in color from l to r. Since the arcs in A^i are only used as bridge between two different colors, they have zero cost. In Figure 3.9 it is shown an example with three colors, whose ordering is the same in Figure 3.5.a.

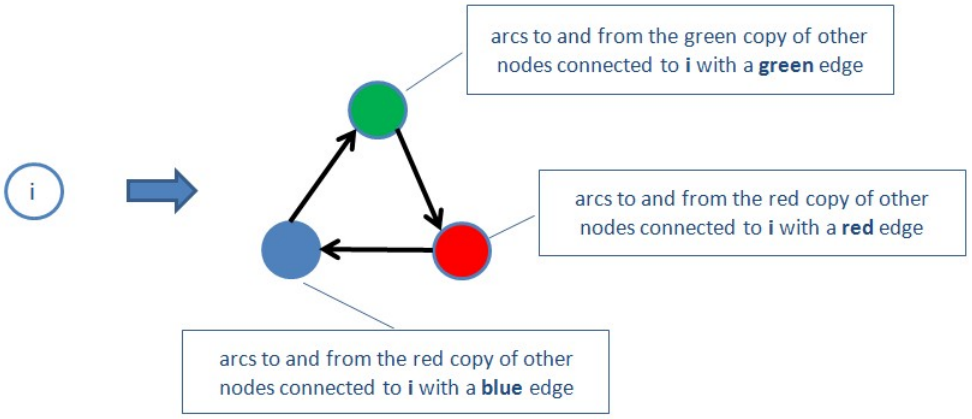


FIGURE 3.9: The color connected component of a vertex of G into the cycles with 3 vertices and 3 arcs.

- the subset A'', whose arcs connect vertices that belong to two partition V^i and V^j, where $i \neq j$. Namely, each original edge $(v^i, v^j) \in E$ with color $r \in \{1, \ldots, c\}$ is represented by two oppositely directed arcs $(v_r^i, v_r^j), (v_r^j, v_r^i) \in A''$ introduced between vertices with color r in V^i and V^j. We collect such arcs within the set A^c, to which we also add arcs connecting s with each vertex v_1^i $(i = 1, \ldots, n)$, and arcs connecting all vertices of V' with the destination vertex t.

3. Each edge $(v^i, v^j) \in E$ with color label $l \in C$ is replaced by two arcs in A'', (v_l^i, v_l^j) and (v_l^j, v_l^i). Therefore, the set A'' only contains arcs with one end in a connected subgraph and the other end in another connected subgraph.

   Both arcs have the same cost c_{ij} from the edge $(v^i, v^j) \in E$. E.g., if $c = 3$ then all pairs of arcs (v_1^i, v_1^j) and $(v_1^j, v_1^i) \in A''$ are from those edges $(v^i, v^j) \in E$ with color

1 in G; all pairs of arcs (v_2^i, v_2^j) and $(v_2^j, v_2^i) \in A''$ are from those edges $(v^i, v^j) \in E$ with color 2 in G, and all pairs of arcs (v_3^i, v_3^j) and $(v_3^j, v_3^i) \in A''$ are from those edges $(v^i, v^j) \in E$ with color 3 in G.

4. One source vertex s and one destination vertex t are added to D, thus $|V'| = n \times c + 2$. Arcs with zero cost connect s with each vertex $v_1^i, i = 1, \ldots, n$, and all vertices of V' to t (the latter referred to as *exit arcs*). Also these arcs belong to A''.

Example 3.3. *Let us consider the same example of the previous sections. The resulting network $D = (V', A)$ has 20 vertices. Each original vertex is represented by a cycle of 3 colored vertices. Two vertices of color c_i are connected by two arcs in opposite directions only if in the original graph they are connected by an edge of color c_i. The source vertex is connected to all vertices of color green; exit arcs to sink are not represented in Figure 3.10 but they are considered for all vertices.*

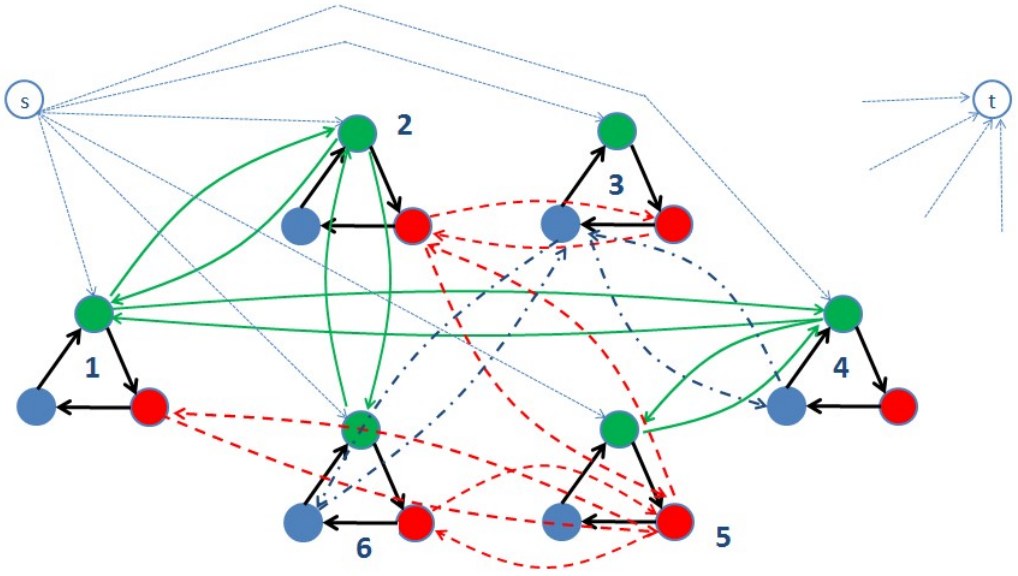


FIGURE 3.10: The 3-connected graph arising from the edge-colored graph in Figure 3.5

Let us describe the corresponding IP formulation of LPCPP. Let $D = (V', A)$ be the directed graph depicted above. Since in this model there are two different set of arcs, we use two types of decision variables: for each arc $(v_l^i, v_l^j) \in A''$, where $v_l^i \in V^i$ and $v_l^j \in V^j$, we refer to x_l^{ij}; for each arc $(v_l^i, v_r^i) \in A^i$ we refer to x_{lr}^i. Both decision variables and arc costs are binaries, and with $c_l^{ij} = 1$ we refer to the cost of all the arcs in A''. The longest path over cyclic c-connected graph (LPCPP) is formulated as follows:

Maximize $\sum_{(v_l^i,v_l^j)\in A''} c_l^{ij} x_l^{ij}$

subject to:

$$\sum_{(v_l^j,v_l^i)\in A''} x_l^{ji} - \sum_{(v_l^i,v_r^i)\in A'} x_{lr}^i = 0 \qquad \forall v_l^i \in V' - \{s,t\} \qquad (C1)$$

$$\sum_{(v_r^i,v_l^i)\in A'} x_{rl}^i - \sum_{(v_l^i,v_l^j)\in A''} x_l^{ji} = 0 \qquad \forall v_l^i \in V' - \{s,t\} \qquad (C2)$$

$$\sum_{(s,v_1^j)\in A''} x_1^{sj} = 1 \qquad (C3)$$

$$\sum_{(v_l^i,t)\in A''} x_l^{it} = 1 \qquad (C4)$$

$$\sum_{(v_l^i,v_r^i)\in A^i} x_{lr}^i \leq 1 \qquad i = 1,2,\ldots,n \qquad (C5)$$

$$\sum_{(v_l^i,v_l^j)\in A''} x_l^{ij} \leq 1 \qquad i = 1,2,\ldots,n \qquad (C6)$$

$$\sum_{(v_l^j,v_l^i)\in A''} x_l^{ji} \leq 1 \qquad i = 1,2,\ldots,n \qquad (C7)$$

$$\sum_{(v_l^i,v_l^j)\in\Gamma} x_l^{ij} + \sum_{(v_l^i,v_r^i)\in\Gamma} x_{lr}^i \leq |\,\Gamma\,| - 1 \qquad \Gamma \in \widehat{\Gamma} \qquad (C8)$$

$$x_l^{ij} \in \{0,1\} \qquad \forall\,(v_l^i,v_l^j) \in A'' \qquad (C9)$$

$$x_{lr}^i \in \{0,1\} \qquad \forall\,(v_l^i,v_r^i) \in A' \qquad (C10)$$

Recalling that arcs internal to the connected subgraphs have zero cost, in the objective function only appear the variables corresponding to the arcs in A''. The constraints (C1-C4) ensure that the solution is a path from s to t. In particular, the constraints (C1-C2) state that anytime the path reaches a vertex v_l^i through an arc with color l, then the next arc in the path must connect it to another vertex v_r^i belong to the same color connected component D^i, which is in turn the head of an arc with color r. In this way the path follows the required ordering of colors. In this model three kinds of *packing constraints* (C5-C7) have been formulated. These constraints ensure that in any color connected component D^i at most one arc can be selected (C5) and at most two vertices can be visited by the path (C6-C7). Such as in LPcPP the graph is not acyclic, thus we need to enforce the separations of all the cycles adding the constraints (C8), that for each cycle $\Gamma \in \widehat{\Gamma}$ express the corresponding cycle elimination constraint.

Preposition 3.3. *A feasible (optimal) solution x^* of LPCPP in D is an orderly colored (longest) path in G. If x^* is composed by $n-1$ edges, then it is an OCHP in G.*

3.4.4 OCLP problem and Shortest Path problems

Although the difference between a path coloring problem and the OCLP problem is quite evident, that one between an existing label graph problem and the OCLP problem is more subtle. One might think that the OCLP problem can be reduced to an elementary shortest path problem, setting negative edge weights, and then solved by using a labeling algorithm or with a dynamic programming approach [63]. For instance, our OCLP problem formulations could appear similar to the Shortest Path Tour Problem (SPTP)

described in [60]. SPTP is a polynomial-time reduction into a s-t shortest path problem of a multi-stage digraph such that the path should successively pass through at least one node from given partite sets of vertices. Nevertheless, the substantial difference between SPTP and OCLP is to be found in the formulation of the constraints on the partite sets: in the former they are covering constraints, while in the latter they are packing constraints. Moreover, the SPTP formulation requires that all arc lengths are nonnegative, so that no negative cycles could hold in the solution. On the other hand, one could guess that OCLP problem can be reduced to an elementary shortest path problem with resource constraints [56, 58], which is solvable by correcting labeling algorithms working even in presence of negative cycles. The requirement that paths need to be properly colored could be fulfilled extending the labels by an additional attribute, which is the color of the last arc. Dominance would be then performed only among labels of the same color, and the path extension step would respect the given order of colors. Unfortunately, this reduction does not guarantee to find a path which visits the largest number of vertices of the graph. Let us recall that in the ideal case we would find an hamiltonian path.

3.4.5 Computational complexity analysis

The Hamiltonian path problem is one of the most well-known NP-complete problems, with numerous applications. The most natural optimization version of this problem is the longest path problem, that is, to compute a simple path of maximum length or, equivalently, to find a maximum induced subgraph which is Hamiltonian. Even if a graph itself is not Hamiltonian, it makes sense in an assignment pathway problem to search for a longest path. However, computing a longest path seems to be more difficult than deciding whether or not a graph admits a Hamiltonian path. Indeed, it has been proved that even if a graph is Hamiltonian, the problem of computing a path of length $n - n^{\epsilon}$ for any $\epsilon < 1$ is NP-hard, where n is the number of vertices of the input graph [82]. Moreover, there is no polynomial time constant-factor approximation algorithm for the longest path problem unless P = NP [82].
In contrast to the Hamiltonian path problem, for which many polynomial time algorithms have been developed with considerable success, there is only a small class of algorithms for solving the longest path problem. To our knowledge, these were restricted to multipartite digraphs [71], trees [35],weighted trees and block graphs [133], bipartite permutation graphs [134], interval graphs [79], and cocomparability graphs [101].

As regards longest path on edge-colored graphs, in literature there are several works showing that to find a properly colored path is NP-hard [1, 17, 43, 69]. Nevertheless, by imposing a few ad-hoc characterization on the edge-colored graph it is possible to

determine a properly colored path (or just to check its existence) in polynomial time, by trying to visit each vertex of the graph the least number of times. This has been proved for several types of problem: longest path [14], path between two given vertices [1, 43, 69, 100], hamiltonian path [15, 28] and cycles [93, 144].
In general, It is well known that alternating edge-colored hamiltonian paths can be found efficiently in 2-edge-colored complete graphs, but it is a long standing question whether there exists a polynomial algorithm for finding such hamiltonian paths in edge-colored complete graphs with three colors or more [17].

Theorem 3.3. *The problem of finding an orderly colored longest path in an edge-colored graph is NP-hard.*

Our proof is based on the reduction of LPnPP from the Longest path problem (LPP), which is well-known being NP-hard [65]. Since for both LPcPP and LPCPP models the network is not acyclic, proving that OCLP problem is NP-hard in these cases is rather trivial; instead, the network representing the LPnPP is directed and acyclic, thus one might think that this problem may be reduced to a shortest path problem which is polynomial solvable. Below we provide a sketch of the proof, which shows that LPnPP is still Np-hard due to the presence of the packing constraints.

Proof. We first rely on the fact that a correct model for OCLP is provided by LPnPP. Then, we note that any LPP instance $G = (V, E)$ can be converted into one of the LPnPP $D = (V', A)$ simply by replicating n times the vertices of G, and by forming n-partitions of the vertex set V'. For each partition composed by n-copies of the same vertex $v \in V$ of G we add a packing constraint. Direct arcs are added between the levels according to the rules described in Section 3.4.1, with the only difference that, in this case, the color constraints on the edges are not present and thus all arcs are replicated between each pair of consecutive levels. The network so obtained is an instance of the LPnPP, and can be solved finding the longest path between two vertices with no cycles. Clearly the transformation of a LPP instance into one of the LPnPP can be performed in polynomial time in the size of the original graph.
Let us now consider the conversion of any LPnPP into a longest path problem. For each partition of the LPnPP instance D, we can remove the associated packing constraints and add two direct arcs which connect, in both directions, each pair of vertices in the same partition. Each of these arcs is associated with a very large positive weights in the objective function. In this way, a great number of augmenting cycles are introduced in the graph, each of which touches two or more vertices in the same partition. Thus, the longest path on such graph contains a cycle every time that two vertices in the same partition are chosen. Conversely, if the longest path on this graph does not contain

cycles, then it is longest with respect to the weights of the edges of D, and this path is in turn a solution of the associated LPnPP. Since also the transformation of LPnPP instance into an LPP instance can be performed in polynomial time in the size of the original graph, we can then conclude that LPnPP belongs to the same complexity class of LPP. This completes the sketch of proof. □

3.5 The assignment pathway procedure at glance

On the basis of the graph models described in Section 3.3 we have developed a procedure solving the 3D-APP. This procedure has been implemented in C programming language and runs in Unix as well as Windows environment.
All the spectral parameters are listed in a text file representing the 3D NMR spectrum after peak-picking procedure. The file specifies all the cross-peaks contained in the spectrum. For each cross-peak, there are: its number, three coordinates (x^i, y^i, z^i) given in ppm or Hz, and widths in three dimensions given in Hz. Additionally, user provides the type of interaction (homo- or heteronuclear), the type of correlation signal (only for a heteronuclear interaction), the type of IP model (see Section 3.4), and the time-limit. A more detailed description of a 3D NMR instance shall be given in Section 3.6.2.1.
In the first step an algorithm reads the input files and constructs an edge-colored graph assigning to any edge an appropriate label according to the scheme in Section 3.3.2. Not-labeled edges are consider incorrect and they are not added to the graph structure, together with all isolated vertices (e.g., vertices connected to others by not-labeled edges). Next, the procedure builds a network representation of the edge-colored graph according to the IP model chosen. Then a mixed integer commercial solver for finding the longest path on the network is called.
The number of possible assignment pathways and their lengths depend on RNA structure [129]. Usually there exist several pathways that satisfy all the required conditions. We assumed that in the first tests of the method all the possible solutions should be returned. Therefore for enumerating all the orderly colored paths of maximum length, the method solve the problem iteratively, by cutting out at each iteration the current optimal solution, until one between the stopping criterions is satisfied. Such criterions are the maximum time-limit, and the length of the path (i.e., when the current solution found is shorter than previous one). Figure 3.11 presents the general view of the method.

3.5.1 A Branch & Cut approach

Last two formulations presented in Section 3.4.2 and Section 3.4.3 rely on the satisfaction of exponentially many cycle elimination constraints, (C5) and (C8) respectively. Such

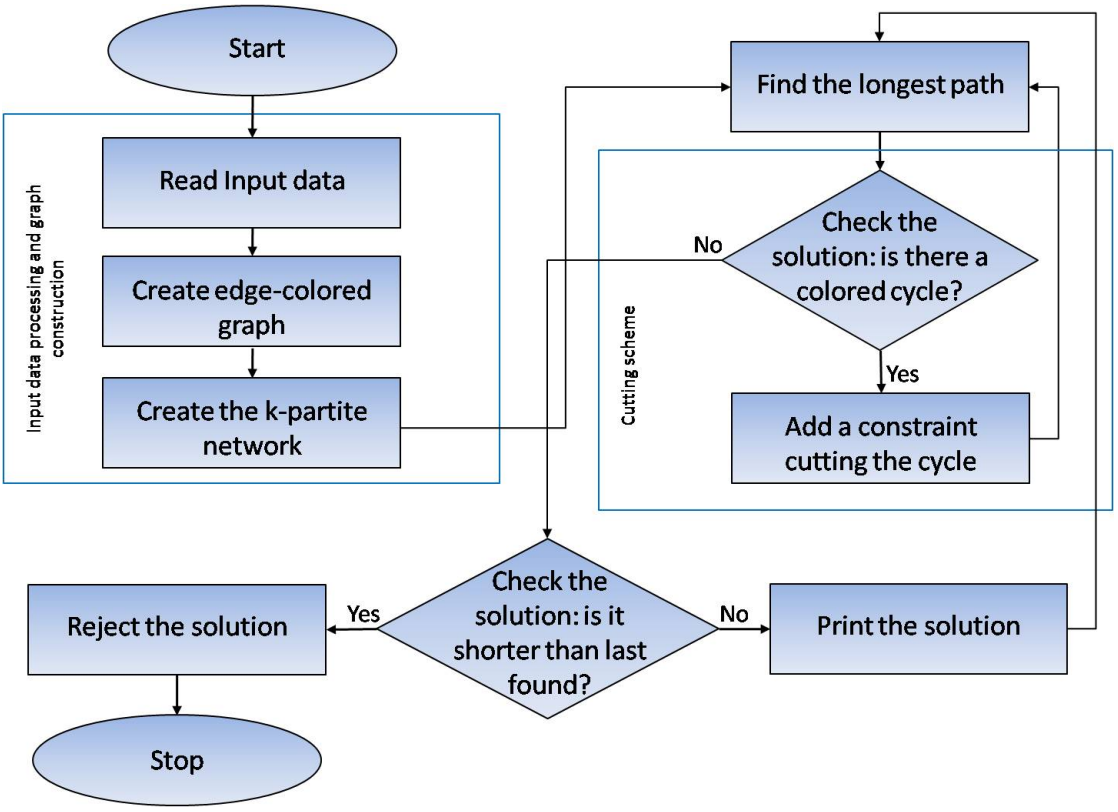


FIGURE 3.11: The flow-chart of the assignment procedure.

constraints cannot be listed explicitly in the formulation and need to be treated with some specific technique (this concerns the cutting scheme in Figure 3.11). Initially, we adopted a simple strategy to deal with them [49]: the cycle elimination constraints were relaxed in the formulation, the relaxed IP was solved and, if a cycle appeared in the integer solution, the corresponding cycle elimination constraint was added to the formulation, and the problem was re-optimized. According to this strategy, the first optimal solution without cycles is the desired optimal solution of the problem. However, we assumed that all the solutions of maximum length should be returned. In this way we have been able to determine the cardinality of the set of the optimal solutions, in order to assess the quality of the reconstructed path from the biochemical point of view. Also in this case, equivalent optimal solutions were listed by cutting out the current optimal solution found, and then re-optimizing the problem. The two models performed reasonably well on problems with a limited number of cross-peaks when a mixed integer commercial solver is used.

At a later stage, we proposed a Branch & Cut (B&C) scheme where a separation oracle is applied to find cycles violated by the fractional solutions at hand in the vertices of the branching tree [127]. In this way, the integer optimization software adopted seeks a solution of the problem by branch-and-cut until it either finds a feasible solution

without orderly colored cycles inside, or shows that one none exists. Below we describe the polynomial separation problem.

The separation procedure is applied straight-forwardly to both LPcPP and LPCPP formulations presented above. Thus, without loss of generality, we consider the graph $D = (V', A)$ associated either to Model 2 (Section 3.4.2) or Model 3 (Section 3.4.3), and we denote with x a binary solution vector of one considered model. We refer with x^a to the variable $x^{ij} \in \{0, 1\}$, for each $a = (i, j) \in A$. Thus, if the arc $a \in A$ is part of the solution x, then $x^a = 1$; otherwise $x^a = 0$.
Given a solution $\tilde{x}$, we define the set $W \subseteq V'$ composed by all vertices that are endpoints of arcs included in $\tilde{x}$: $W = \{i, j \in V' : \tilde{x}^a = 1, a = (i, j)\}$.
Let now $z \in \{0, 1\}^{|W|}$ be the incidence vector of W, $A[W]$ be the set of edges connecting vertices in W, and $y \in \{0, 1\}^{|A[W]|}$ be the incidence vector of $A[W]$. We first note that a subset $\Gamma \subseteq W$ identifies a cycle in V' if $\sum_{i \in \Gamma, j \in \Gamma} \tilde{x}^{ij} > |\Gamma| - 1$. With the aid of the y variables just defined, we can write the following linear integer problem:

$$
\begin{aligned}
&\text{Maximize} \quad \textstyle\sum_{a \in A[W]} \tilde{x}^a y^a - \sum_{i \in W} z_i \\
&\text{subject to:} \\
&\qquad y^a \le z_i && \forall a \in A[W], \quad a = (i, j) \\
&\qquad y^a \le z_j && \forall a \in A[W], \quad a = (i, j) \\
&\qquad y^a \ge z_i + z_j - 1 && \forall a \in A[W], \quad a = (i, j) \\
&\qquad y^a \in \{0, 1\} && \forall a \in A[W] \\
&\qquad z_i \in \{0, 1\} && \forall i \in W
\end{aligned}
\tag{3.1}
$$

The problem (3.1) handles vertices and edges of the solution vector $\tilde{x}$. The first two constraints guarantee that for each arc $a \in A[W]$ considered in the solution of (3.1) the endpoints (i, j) must have associated $z_i = 1$ and $z_j = 1$ respectively. The third constraint states that if two vertices $i, j \in W$ are endpoints of arcs included in the solution of (3.1) then the arc $a = (i, j)$ connecting them must have associated $y^a = 1$. In oher words, if an arc $a = (i, j)$ is included in the solution of (3.1), then its endpoints are also included. When the optimal solution $(\bar{z}, \bar{y})$ has objective function value grater than -1 it means that the number of arcs is higher than the number of vertices, therefore such solution identifies a cycle in the original solution $\tilde{x}$. Conversely, if no solution exists, we conclude that no cycle is present in the solution represented by $\tilde{x}$.
Previous results (cf. [140]) showed that problem (3.1) has integer optimal solution also when the integrality constraints on the z and y variables are relaxed. The linear relaxation of (3.1) is thus a polynomial separation oracle that can be applied at each vertex of the B&C tree.

3.6 Computational results

In order to test the strength of our models from both mathematical and biochemical point of view, we conducted two different batteries of tests. In Section 3.6.1 is presented a battery of test conducted on randomly generated edge-colored graphs and presented in [128]. In Section 3.6.2 is presented a battery of test conducted on a set of instances representing 3D NMR spectrum and presented in [127]. Let us recall that in this latter case the procedure include the branch-and-cut scheme described in Section 3.5.1. In both cases the results have shown that our procedure solves efficiently the problem, by finding the optimal solution known a priori.

The experiments were run using Mixed Integer Linear Programming solver Cplex 12.2.0 by IBM Ilog [80] with standard settings. The hardware platform was an 8-core i7 Intel processor 2.597GHz with 8GB RAM. The code was developed in C programming language and compiled with GNU CC compiler running under Microsoft Windows 7.0 with optimization option O3. The cycle separation procedure (cf. Section 3.5.1) was implemented as a user callback function. It was managed by the main call to the mixed integer optimizers. Presolve reductions were switched off and callback were set to access the non-reduced model. The traditional MIP search strategy was used, and the solver was run as a single thread.

3.6.1 Test on edge-colored graphs

In this section we show how large instances of OCLP problem can be solved in an efficient way using the proposed IP models. We describe the instance generator developed for creating the set of instances used during our test, and then we analyze and discuss the results of computational experiments.

3.6.1.1 Instance simulator

The instance generator described here generates an c-edge-colored graph without taking into account the biochemical information of an 3D NMR map. In this way we only test the ability of our IP models to find the set of optimal solutions.
The OCLP problem instance generator produces a `DIMACS` Graph Format file, which is a flexible format suitable for many types of graph and network problems. This format was also chosen by *Rutgers University* for the First Computational Challenge on network flows and matchings[3]. In the dimacs file all the edges of the c-edge-colored graph and their correspondent color labels are listed.

[3] http://prolland.free.fr/works/research/dsat/dimacs.html

The instance generator has been designed to generate a c-edge-colored graph $G = (V, E)$ where $|E|$ depends on graph density d and number of vertices n set such that $|E| = n(n-1)d/2$. The three main parameters governing the generation are (i) the number of vertices n, (ii) the density of the graph d, and (iii) the number of OCPs M. The user can also provide a sequence of two or three colors (OCSeq).
Moreover, the generator allows an additional control step related with the presence of an OCHP; thus, we generate two types of problems:

- In problems of *type 1* the OCPs in the generated graph may have any length L, up to $n-1$;
- In problems of *type 2* the generation procedure is designed to create at least one OCHP.

A sketch of this simulator is given below.

c-edge-colored graph generation

while $(i < M)$ **do**
  $V = \{s\}$ and $E = \emptyset$;
  while $(|E| < L)$ **do**
    generate the vertex $v \notin V$ such that $v \in \{1, \ldots, n\}$;
    assign to the current edge (s, v) a color r according to the OCSeq;
    if ((s, v) belong to a pre-existing path P_i) **then**
      if (the color of (s, v) is different from r in P_i) **then**
        reject l'arc (s, v);
      else
        $V = V \cup \{v\}$ and $E = E \cup \{(s, v)\}$;
        $s \leftarrow v$;
      end if
    else
      $V = V \cup \{v\}$ and $E = E \cup \{(s, v)\}$;
      $s \leftarrow v$;
    end if
  end while
end while
if (the graph density d has not been reached) **then**
generate the remaining edges with random colors from $\{1, \ldots, c\}$
end if

3.6.1.2 Experimental results

A detailed report of the experiments is summarizes in Tables 3.1-3.4, each table is referred to a different set of experiments with graph of increasing dimension (from 20 to 100 vertices) and different arc densities (here 3 levels are considered: 10%, 20%, and 30%). Let us recall that 2-OCLP specifies the problem of finding an alternating longest path, whereas 3-OCLP specifies the problem of finding an OCLP problem whose edges follow a given sequence of three colors. Graphs on which alternating paths were sought are reported in Table 3.1 and Table 3.2, for problems of type 1. Tables 3.3 and Table 3.4, instead, report 3-OCLPs of type 2 (i.e. OCHP problem instances).
Each table reports in the first column the density of the edge-colored graph, followed by the number of its vertices, the number of vertices and arcs (or variables) in the expanded partite graph, the number of constraints in the associated ILP (this number does not include all the constraints added to cut out the current solution, iteratively), the length of the optimal path, the number of paths of maximum length found, and the number of cycles cut out during the computation (for experiments run with model 2 (LPcPP) and model 3 (LPCPP)). The last column reports the total computation time in seconds; when solution time exceeded 1 hour, the algorithm was halted and the current state of the solution printed.

The results reported in Tables 3.1-3.4 bring into evidence a computational challenge when the dimension of the corresponding network reaches reasonable sizes, especially for the computation times required for solution by the three models.
An additional insight in this direction is provided in Table 3.5, where we report the average solution times for the 3 models, for different sizes of the graph (the average is taken over all the dimensions considered, e.g., density, type, number of colors). The table shows a quick rise of solution time related to the number of vertices, and, accordingly, the higher solution times required by Model 1 (LPnPP). Not surprisingly, for large problems the difference in average time for the 3 models tends to reduce, as all the 3 models end up spending the whole hour that is given as upper bound on computation time.

Despite the first model is the only one finding the optimal solutions with no cycle inside, when the graph is denser it often fails in finding an optimal solution within the time bound, differently from Model 2 and Model 3. As it is shown in Table 3.6, where a comparison between the three models on 36 problems of large size (with 50, 70 and 100 vertices) is provided.

From all these results we have guessed a superiority of Model 2 and Model 3 (LPcP and LPCPP) over Model 1 (LPnPP). To further reinforce this consideration, we compared the three models on a set of 10 randomly generated problems with an interesting degree

Graph Density	Cross Peaks	Model	Vertices	Arcs (var.)	N. of Constr.	Path Length	N. of Paths	N. of Cycles	Total Time (s)
	20	1	402	653	423	7	1	0	0
0.1	20	2	41	64	62	7	1	0	0
	20	3	41	104	141	7	1	0	0
	20	1	402	689	423	7	1	0	0
0.2	20	2	41	68	62	7	1	0	0
	20	3	41	108	141	7	1	0	0.02
	20	1	402	827	424	7	2	0	0.02
0.3	20	2	41	82	63	7	2	0	0.02
	20	3	41	122	142	7	2	0	0.02
	30	1	902	1513	933	6	1	0	0
0.1	30	2	61	100	92	6	1	0	0
	30	3	61	160	211	6	1	0	0
	30	1	902	1949	936	7	4	0	0.08
0.2	30	2	61	130	97	7	4	2	0.03
	30	3	61	190	216	7	4	2	0.06
	30	1	902	2327	936	11	4	0	0.25
0.3	30	2	61	156	95	11	4	0	0.06
	30	3	61	216	214	11	4	0	0.08
	50	1	2502	4427	2553	9	1	0	0.03
0.1	50	2	101	176	152	9	1	0	0
	50	3	101	276	351	9	1	0	0
	50	1	2502	6429	2553	19	1	0	1.08
0.2	50	2	101	258	154	19	1	2	0.05
	50	3	101	358	353	19	1	2	0.11
	50	1	2502	8535	2660	34	108	0	553.46
0.3	50	2	101	344	366	34	108	107	24.4
	50	3	101	444	564	34	108	106	36.29
	70	1	4902	22743	4972	69	0	0	>3600
0.1	70	2	141	656	211	69	325	1022	>3600
	70	3	141	796	490	69	227	842	>3600
	70	1	4902	22743	4972	69	0	0	>3600
0.2	70	2	141	656	211	69	237	1418	>3600
	70	3	141	796	490	69	189	1231	>3600
	70	1	4902	22743	4972	69	9		>3600
0.3	70	2	141	656	211	69	736	1996	>3600
	70	3	141	796	490	69	737	1961	>3600
	100	1	10002	28013	10107	33	5	0	40.08
0.1	100	2	201	562	306	33	5	0	0.25
	100	3	201	762	705	33	5	0	0.27
	100	1	10002	43457	10102	N.F.	0	0	>3600
0.2	100	2	201	874	2535	96	509	1725	>3600
	100	3	201	1074	2658	96	430	1528	>3600
	100	1	10002	63953	10102	N.F.	0	0	>3600
0.3	100	2	201	1288	3871	99	136	3434	>3600
	100	3	201	1488	4197	99	163	3334	>3600

TABLE 3.1: Experimental results for LPnPP, LPcPP, and LPCPP models, for 2-OCLPs of type 1 (no injected OCHP).

of difficulty. All problems presented the same characteristics: 2 colors, 100 vertices, density 20%, no injected Hamiltonian path.

The results in Table 3.7 have confirmed our hypothesis. Not surprisingly, all problems exploited all the allotted time. Model 2 determined, on average, a larger number of solutions than Model 3, by working a little harder in cutting out cycles.

From the comparison among each of the 10 problems according to the number of optimal solutions and of the eliminated cycles found (see Figure 3.12) seems that Model 2 and Model 3 have similar performances.

Graph Density	Cross Peaks	Model	Vertices	Arcs (var.)	N. of Constr.	Path Length	N. of Paths	N. of Cycles	Total Time (s)
	20	1	402	641	423	5	1	0	0
0.1	20	2	61	94	82	5	1	0	0
	20	3	61	154	181	5	1	0	0
	20	1	402	725	423	5	1	0	0
0.2	20	2	61	108	83	5	1	1	0.02
	20	3	61	168	182	5	1	1	0
	20	1	402	881	428	8	6	0	0.05
0.3	20	2	61	132	88	8	6	1	0.05
	20	3	61	192	187	8	6	1	0.05
	30	1	902	1515	933	5	1	0	0.02
0.1	30	2	91	150	123	5	1	1	0
	30	3	91	240	272	5	1	1	0
	30	1	902	1921	933	10	1	0	0.02
0.2	30	2	91	192	123	10	1	1	0
	30	3	91	282	272	10	1	1	0.03
	30	1	902	2363	936	17	4	0	0.23
0.3	30	2	91	238	127	17	4	2	0.08
	30	3	91	328	276	17	4	2	0.14
	50	1	2502	4779	2554	5	2	0	0.03
0.1	50	2	151	286	205	5	2	2	0
	50	3	151	436	454	5	2	2	0.02
	50	1	2502	6765	2553	33	1	0	1.73
0.2	50	2	151	408	205	33	1	3	0.08
	50	3	151	558	454	33	1	3	0.08
	50	1	2502	8883	2560	44	8	0	>3600
0.3	50	2	151	536	314	44	34	79	70.03
	50	3	151	686	563	44	34	79	101
	70	1	4902	21485	4972	N.F.	0	0	>3600
0.1	70	2	211	928	688	69	18	122	>3600
	70	3	211	1138	1034	69	16	107	>3600
	70	1	4902	21485	4972	N.F.	0	0	>3600
0.2	70	2	211	928	688	69	28	346	>3600
	70	3	211	1138	1034	69	22	229	>3600
	70	1	4902	21485	4972	N.F.	0	0	>3600
0.3	70	2	211	928	688	69	51	356	>3600
	70	3	211	1138	1034	69	58	346	>3600
	100	1	10002	26337	10106	47	4	0	764.47
0.1	100	2	301	792	412	47	4	7	2.03
	100	3	301	1092	911	47	4	7	3.23
	100	1	10002	44619	10102	N.F.	0	0	>3600
0.2	100	2	301	1346	1087	97	53	633	>3600
	100	3	301	1646	1515	98	47	568	>3600
	100	1	10002	58545	10102	N.F.	0	0	>3600
0.3	100	2	301	1768	1567	99	35	1131	>3600
	100	3	301	2068	2354	99	61	1393	>3600

TABLE 3.2: Experimental results for LPnPP, LPcPP, and LPCPP models for 2-OCLPs of type 1 (no injected OCHP).

3.6.2 Test on NMR data

Since the superiority in terms of time of the last two IP formulations, we used only this two models to test the efficiency of these two models on 3D-APP instances. We also studied the improvement of the methods when the B&C oracle is applied. To do so, we used test instances simulated by an instance simulator developed ad hoc, of which a description is provided below. Then we analyze and discuss the results of computational experiments.

Graph Density	Cross Peaks	Model	Vertices	Arcs (var.)	N. of Constr.	Path Length	N. of Paths	N. of Cycles	Total Time (s)
	20	1	402	783	423	19	1	0	0
0.1	20	2	41	78	62	19	1	0	0
	20	3	41	118	141	19	1	0	0
	20	1	402	895	423	19	1	0	0.03
0.2	20	2	41	90	62	19	1	0	0.02
	20	3	41	130	141	19	1	0	0.03
	20	1	402	1035	423	19	1	0	0.03
0.3	20	2	41	104	62	19	1	0	0.02
	20	3	41	144	141	19	1	0	0.02
	30	1	902	1923	933	29	1	0	0.02
0.1	30	2	61	128	92	29	1	0	0.02
	30	3	61	188	211	29	1	0	0.02
	30	1	902	2357	933	29	1	0	0.51
0.2	30	2	61	158	92	29	1	0	0.05
	30	3	61	218	211	29	1	0	0.03
	30	1	902	2767	934	29	2	0	0.69
0.3	30	2	61	186	97	29	2	4	0.2
	30	3	61	246	216	29	2	4	0.19
	50	1	2502	5737	2553	49	1	0	2.03
0.1	50	2	101	230	152	49	1	0	0.05
	50	3	101	330	351	49	1	0	0.08
	50	1	2502	8089	2558	49	6	0	5.16
0.2	50	2	101	326	161	49	6	4	0.53
	50	3	101	426	351	0	1	4	1.2
	50	1	2502	8089	2558	49	22	0	328
0.3	50	2	101	326	161	49	22	436	97
	50	3	101	426	351	0	22	436	124
	70	1	4902	19937	4972	0	0	0	>3600
0.1	70	2	141	574	211	69	124	768	1425
	70	3	141	714	490	69	124	894	1829
	70	1	4902	19937	4972	0			>3600
0.2	70	2	141	574	211	69	654	2276	>3600
	70	3	141	714	490	69	559	2009	>3600
	70	1	4902	26783	4972	0			>3600
0.3	70	2	141	772	211	69	257	2820	2206
	70	3	141	912	490	69	257	2975	2341
	100	1	10002	33279	10122	99	20	0	454.98
0.1	100	2	201	668	341	99	20	20	6.54
	100	3	201	868	741	99	20	21	12.9
	100	1	10002	51865	10102	N.F.	0	0	>3600
0.2	100	2	201	1044	3412	99	151	2960	>3600
	100	3	201	1244	3755	99	162	2893	>3600
	100	1	10002	66155	10102	N.F.	0	0	>3600
0.3	100	2	201	1332	3983	99	143	3539	>3600
	100	3	201	1532	4358	99	123	3535	>3600

TABLE 3.3: Experimental results for LPnPP, LPcPP, and LPCPP models for 3-OCLPs of type 2 (injected OCHP).

3.6.2.1 Instance simulator

As already pointed out in Section 3.2.1, NMR spectroscopy has been a well established technique to study the structures of biological molecules. The evaluation of protein and RNA structures obtained from NMR proves, that two-dimensional experiments are not sufficient to compute high-quality models of large molecules [99]. Thus, the need to elucidate and analyze large molecules results in increasing the dimensionality of NMR experiments.

Since the 3D NMR for RNA has been introduced relatively late - as compared to the

Graph Density	Cross Peaks	Model	Vertices	Arcs (var.)	N. of Constr.	Path Length	N. of Paths	N. of Cycles	Total Time (s)
	20	1	402	663	423	19	1	0	0
0.1	20	2	61	98	82	19	1	0	0
	20	3	61	158	181	19	1	0	0
	20	1	402	763	423	19	1	0	0
0.2	20	2	61	114	82	19	1	0	0
	20	3	61	174	181	19	1	0	0.03
	20	1	402	921	423	19	1	0	0.03
0.3	20	2	61	138	82	19	1	0	0
	20	3	61	198	181	19	1	0	0.02
	30	1	902	1669	933	29	1	0	0
0.1	30	2	91	166	122	29	1	0	0.02
	30	3	91	256	271	29	1	0	0.03
	30	1	902	2117	933	29	1	0	0.05
0.2	30	2	91	212	122	29	1	0	0.03
	30	3	91	302	271	29	1	0	0.03
	30	1	902	2643	933	29	1	0	0.45
0.3	30	2	91	266	122	29	1	0	0.05
	30	3	91	356	271	29	1	0	0.06
	50	1	2502	5417	2553	49	1	0	0.3
0.1	50	2	151	326	202	49	1	0	0.03
	50	3	151	476	451	49	1	0	0.03
	50	1	2502	7321	2553	49	1	0	2.04
0.2	50	2	151	442	202	49	1	0	0.14
	50	3	151	592	451	49	1	0	0.17
	50	1	2502	7321	2553	49	16	0	822
0.3	50	2	151	442	202	49	16	87	352
	50	3	151	592	451	49	16	87	528
	70	1	4902	23141	4972	N.F.	1	0	34
0.1	70	2	211	1000	1330	69	1	422	8
	70	3	211	1210	2516	69	1	422	2
	70	1	4902	23141	4972	N.F.	0	0	>3600
0.2	70	2	211	1000	1330	69	48	642	1824
	70	3	211	1210	2516	69	23	328	>3600
	70	1	4902	23141	4972	N.F.	0	0	>3600
0.3	70	2	211	1000	1330	69	78	971	>3600
	70	3	211	1210	2516	69	166	1720	>3600
	100	1	10002	30693	10103	99	1	0	17.08
0.1	100	2	301	924	402	99	1	0	0.45
	100	3	301	1224	901	99	1	0	0.59
	100	1	10002	45081	10102	N.F.	0	0	>3600
0.2	100	2	301	1360	905	99	18	486	>3600
	100	3	301	1660	1660	99	33	727	>3600
	100	1	10002	60525	10102	N.F.	0	0	>3600
0.3	100	2	301	1828	1552	99	27	1124	>3600
	100	3	301	2128	2854	99	70	1884	>3600

TABLE 3.4: Experimental results for LPnPP, LPcPP, and LPCPP models for 3-OCLPs of type 2 (injected OCHP).

Number of Vertices	Model 1	Model 2	Model 3
10	0.005	0.006	0.003
20	0.013	0.011	0.016
30	0.193	0.045	0.056
50	505.822	45.359	65.915
70	3303.272	2765.605	3053.289
100	2506.478	2402.853	2404.793
All Problems	1052.631	868.980	920.679

TABLE 3.5: Solution time (secs.) for LPnPP, LPcPP, and LPCPP; average over all problems, by size.

Graph Density	Model 1	Model 2	Model 3
0.1	9	12	12
0.2	7	12	12
0.3	7	12	12
All Problems	23	36	36

TABLE 3.6: Number of solved problems (problems where at least 1 optimal solution is found within 3600 seconds of computation) for LPnPP, LPcPP, and LPCPP, by graph density; analysis limited to problem with 50, 70, 100 vertices.

Model	Number of solved problems*	Average number of long. paths	Average number of cycles
1	2	1	0
2	10	207.3	2855.6
3	10	196.1	2703.3

TABLE 3.7: Performance comparison over 10 instances of 100 vertices, type 1, for 2-OCLPs on graph with d = 0.2. *At least one optimal solution.

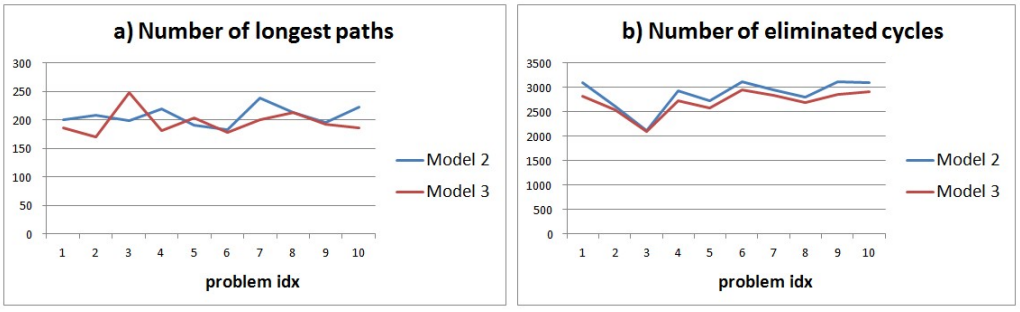


FIGURE 3.12: Number of optimal solutions (a) and eliminated paths (b) for 10 randomly generated problems of large size (100 vertices, 0.2 graph density, 2 colors, no Hamiltonian path injected). Analysis limited to Models 2 and 3 (LPcPP and LPCPP).

protein field - not much experimental data for already solved cases is collected and publicly available for an analysis. Therefore, we implemented an instance generator to simulate the reliable 3D NMR data together with the assignment pathways. This simulator will be publicly available soon, and it can be used for future computational experiments. The presented methods for solving 3D-APP (see Section 3.3.2) was, thus, tested using a set of generated instances, with the number of cross-peaks (instance size) similar to that found in the real NMR experiments.

The 3D-APP instance simulator produces NMR data in a form of a cross-peak list of length n. For each cross-peak p^i in the list, the generator provides its position on the map x^i, y^i, z^i (i.e. three coordinates given in ppm, representing chemical shifts of the involved atoms) and its width for each dimension dx^i, dy^i, dz^i. To simulate the reliable coordinates of cross-peaks, it uses the chemical shift statistics for all atoms included in RNA nucleotides (A,C,G,U) downloaded from the *Biological Magnetic Resonance Data Bank* [135]. Statistics used are reported in Table 3.8.

Nucleus	Residue including nucleus	Chemical shift range
H1'	A,C,G,U	5.31 ÷ 6.10
H4'	A,C,G,U	3.99 ÷ 4.71
H6	C,U	7.46 ÷ 7.96
H8	A,G	7.23 ÷ 8.27
C4'	A,C,G,U	79.57 ÷ 84.63
P	A,C,G,U	-4.82 ÷ -1.41

TABLE 3.8: Statistical data deposited in the Biological Magnetic Resonance Data Bank on 11/01/2012.

The simulator starts from generating the sequence of n cross-peaks with their coordinates and widths. The coordinate values depend on the NMR experiment (heteronuclear HCP or homonuclear NOESY) and RNA sequence, which determine the atoms involved in the simulated interactions. Next, cross-peak widths are used to construct intervals for each coordinate:

$$Ix^i = [x^i - dx^i, x^i + dx^i],$$

$$Iy^i = [y^i - dy^i, y^i + dy^i],$$

$$Iz^i = [z^i - dz^i, z^i + dz^i].$$

This information is next used to create the edge-colored graph $G = (V, E)$ in the first step of our procedure, represented as a list of edges with their colors. For each pair of cross-peaks, $p^i(x^i, y^i, z^i)$ and $p^j(x^j, y^j, z^j)$, on the simulated NMR map, the edge (v^i, v^j) is added to E if at least one of the following conditions is satisfied: $x^i \in Ix^j$, $y^i \in Iy^j$, or $z^i \in Iz^j$. All the edges are colored following the scheme based on the relation between coordinates. To guarantee a certain variability in the set of experiments, an additional parameter $0 \leq \alpha \leq 1$ has been added to the simulator. It represents the probability that current edge color is the same as the next color in the pathway color pattern. For instance, if the color pattern is {*red, green, blue*} and recently generated edge was *red*, then the next edge will be *green* with probability α, or it will have any other color with probability $\frac{1-\alpha}{|C|-1}$, where C is the set of all colors used in the graph. Due to this strategy, problems with longer orderly colored paths correspond to the higher values of α.

3.6.2.2 Experimental results

For the test purposes we generated 50 instances of the 3D-APP problem (25 heteronuclear maps and 25 homonuclear ones) with 5 different instance sizes: 30, 40, 50, 75, 100 cross-peaks, and 5 different values of α parameter: 0.3, 0.4, 0.5, 0.6, 0.7. Each of these 50 generated instances was solved with the relaxation of the last two models described in Section 3.4. For the sake of clarity, we refer to LPcPP as Model 1 and to LPCPP

as Model 2. The B&C approach was used to separate the fractional cycles in the optimal solution. The standard MIP with built-in cuts was applied to separate the integer cycles. Once an optimal solution was found, a constraint was added to the formulation to exclude such solution in the succeeding iterations. Next, the algorithm was iterated. When the optimal solution with smaller value of the objective function was obtained, the iterations were stopped. Each run was anyway stopped after a maximum running time of 3600 seconds.

Each instance was solved in four different ways according to the cycle separation procedures. In the remainder of this section, the tested methods are referred according to the following convention:

(1) `M1-IC`: Model 1 with separation of integer cycles in the optimal solution,

(2) `M2-IC`: Model 2 with separation of integer cycles in the optimal solution,

(3) `M1-FC`: Model 1 with separation of fractional cycles in the B&C scheme,

(4) `M2-FC`: Model 2 with separation of fractional cycles in the B&C scheme.

Table 3.9 and Table 3.10 report the details of the results for Model 1 and Model 2, respectively; in the columns there are the value of the α parameter used for generation, the number of cross-peaks, and for the two variants (with integer and fractional cycle separation) is given the value of the optimal solution, the number of cycles separated and the computing time.

An additional parameter used for comparing the methods is the number of equivalent solutions found in the available time. Moreover, the different methods may have different performances according to problem type and size, although it may not be evident from the analysis of the average results. For this reason, we propose a first analysis based on the definition of a dominance criterion among the methods. We say that method A dominates method B when: (1) both methods find the same number of optimal solutions, but method A requires a shorter total computing time; (2) both methods use all the available time but method A finds more optimal solutions.
We note that methods using FC dominate more often. In particular, the dominance of FC methods grows with the instance size (e.g., if we consider the comparison between M2-FC and M2-IC, we observe that M2-IC dominates M2-FC in problems of small size, but when larger problems are considered, M2-FC becomes more relevant, as shown in Table 3.11). This is particularly evident in Table 3.12, where the number of times each method dominates the others is reported. Let us remark that here the set of instances considered is composed by 32 problems whose solution time is higher than 10 seconds.

			M1-IC			M1-FC		
Exp. Name	value of α	# C.peaks	Opt. Value	# Cycles	Time (secs.)	Opt. Value	# Cycles	Time (secs.)
NOESY-1	0.3	30	3	244	2	3	82	6
NOESY-2	0.4	30	23	45	25	23	10	10
NOESY-3	0.5	30	21	12	1	21	5	0
NOESY-4	0.6	30	23	30	6	23	3	2
NOESY-5	0.7	30	14	210	8	13	23	2
NOESY-6	0.3	40	34	231	48	34	26	32
NOESY-7	0.4	40	30	2211	68	30	103	43
NOESY-8	0.5	40	27	940	120	27	28	38
NOESY-9	0.6	40	34	213	4	34	29	2
NOESY-10	0.7	40	34	422	19	34	47	9
NOESY-11	0.3	50	45	697	453	44	116	473
NOESY-12	0.4	50	47	6815	1830	47	292	102
NOESY-13	0.5	50	46	345	220	46	216	164
NOESY-14	0.6	50	49	356	158	49	273	744
NOESY-15	0.7	50	49	404	514	49	201	650
NOESY-16	0.3	75	74	700	1886	74	19	3600
NOESY-17	0.4	75	73	2526	657	73	94	3061
NOESY-18	0.5	75	74	895	311	74	1204	1658
NOESY-19	0.6	75	73	1052	273	73	58	2136
NOESY-20	0.7	75	74	1072	3600	74	13	3600
NOESY-21	0.3	100	99	850	2163	99	26	3600
NOESY-22	0.4	100	99	654	382	99	14	1911
NOESY-23	0.5	100	99	487	222	99	1821	1774
NOESY-24	0.6	100	99	448	300	98	1645	1684
NOESY-25	0.7	100	98	1272	3600	98	2	3600
HCP-1	0.3	30	26	998	22	26	101	12
HCP-2	0.4	30	20	194	8	20	43	3
HCP-3	0.5	30	0	16	0	0	17	0
HCP-4	0.6	30	24	89	3	24	36	1
HCP-5	0.7	30	26	105	8	26	98	3
HCP-6	0.3	40	33	247	9	33	193	8
HCP-7	0.4	40	35	188	6	35	137	3
HCP-8	0.5	40	32	796	29	32	89	9
HCP-9	0.6	40	24	35	3	24	20	1
HCP-10	0.7	40	34	1779	36	34	245	7
HCP-11	0.3	50	46	13184	3600	46	867	407
HCP-12	0.4	50	46	987	85	46	586	66
HCP-13	0.5	50	42	496	18	42	543	15
HCP-14	0.6	50	44	6329	3600	44	411	71
HCP-15	0.7	50	44	5785	3600	44	580	190
HCP-16	0.3	75	72	5477	3600	72	52	3600
HCP-17	0.4	75	70	6712	3600	70	1527	364
HCP-18	0.5	75	74	658	55	74	1276	206
HCP-19	0.6	75	70	13211	3600	70	91	3600
HCP-20	0.7	75	72	466	54	72	1253	227
HCP-21	0.3	100	99	4899	1814	99	1502	3600
HCP-22	0.4	100	99	2271	1820	99	12	3600
HCP-23	0.5	100	98	741	94	98	3136	1428
HCP-24	0.6	100	98	5947	1913	98	98	3600
HCP-25	0.7	100	0	27	3600	98	16	3600

TABLE 3.9: Results for Model 1

	value of α	# C.peaks	**M2-IC** Opt. Value	# Cycles	Time (secs.)	**M2-FC** Opt. Value	# Cycles	Time (secs.)
NOESY-1	0.3	30	3	82	13	3	244	2
NOESY-2	0.4	30	23	10	21	22	42	12
NOESY-3	0.5	30	21	5	1	21	21	1
NOESY-4	0.6	30	23	3	4	23	25	3
NOESY-5	0.7	30	14	23	4	14	190	4
NOESY-6	0.3	40	34	26	63	34	227	34
NOESY-7	0.4	40	30	104	96	30	2195	36
NOESY-8	0.5	40	27	28	70	26	893	91
NOESY-9	0.6	40	34	26	3	34	252	3
NOESY-10	0.7	40	34	45	14	33	379	10
NOESY-11	0.3	50	45	92	653	45	745	330
NOESY-12	0.4	50	47	215	141	47	3922	148
NOESY-13	0.5	50	46	242	839	46	474	93
NOESY-14	0.6	50	49	335	532	49	811	209
NOESY-15	0.7	50	49	176	593	49	1722	323
NOESY-16	0.3	75	74	60	1920	73	1126	3600
NOESY-17	0.4	75	73	551	3057	73	1072	2514
NOESY-18	0.5	75	74	1187	3078	74	13293	2808
NOESY-19	0.6	75	73	365	2258	73	939	631
NOESY-20	0.7	75	0	21	3600	74	1067	2748
NOESY-21	0.3	100	99	176	3600	98	1002	1911
NOESY-22	0.4	100	99	1886	3600	99	1025	1802
NOESY-23	0.5	100	99	1793	1444	99	1339	501
NOESY-24	0.6	100	99	2023	2109	99	502	285
NOESY-25	0.7	100	0	1	3600	98	1162	2463
HCP-1	0.3	30	26	99	29	26	609	9
HCP-2	0.4	30	20	51	6	20	134	4
HCP-3	0.5	30	0	17	1	0	15	0
HCP-4	0.6	30	24	40	2	24	79	1
HCP-5	0.7	30	26	85	11	26	132	4
HCP-6	0.3	40	33	236	13	33	257	6
HCP-7	0.4	40	35	98	5	35	141	3
HCP-8	0.5	40	32	109	24	32	630	13
HCP-9	0.6	40	24	34	2	24	102	2
HCP-10	0.7	40	34	292	13	34	404	6
HCP-11	0.3	50	46	32	3600	46	903	53
HCP-12	0.4	50	46	406	51	46	333	18
HCP-13	0.5	50	42	652	33	42	328	8
HCP-14	0.6	50	44	139	3600	44	397	18
HCP-15	0.7	50	44	252	3600	44	543	28
HCP-16	0.3	75	0	3	3600	71	4398	3600
HCP-17	0.4	75	70	1843	489	70	451	41
HCP-18	0.5	75	74	1663	338	74	420	35
HCP-19	0.6	75	0	3	3600	70	6201	3600
HCP-20	0.7	75	72	1196	265	72	420	44
HCP-21	0.3	100	99	2686	2691	98	4150	3600
HCP-22	0.4	100	99	2828	3600	98	4316	3600
HCP-23	0.5	100	98	3314	1331	98	597	102
HCP-24	0.6	100	98	2879	3600	97	3638	3600
HCP-25	0.7	100	0	3	3600	98	5162	2599

TABLE 3.10: Results for Model 2

	Instance size					Total
Method	30	40	50	75	100	dominations
M1-FC	3	3	8	4	3	21
M1-IC	7	7	1	1	0	16
M2-FC	0	0	1	5	7	13
M2-IC	0	0	0	0	0	0
Total	10	10	10	10	10	50

TABLE 3.11: Method domination according to instance size (all instances).

Method	Instance size 30	40	50	75	100	Total domination index
M1-FC	0	1	7	4	3	15
M1-IC	0	2	1	1	0	4
M2-FC	0	0	1	5	7	13
M2-IC	0	0	0	0	0	0
Total	0	3	9	10	10	32

TABLE 3.12: Method domination according to instance size, for instances with computing time $\geq$ 10s.

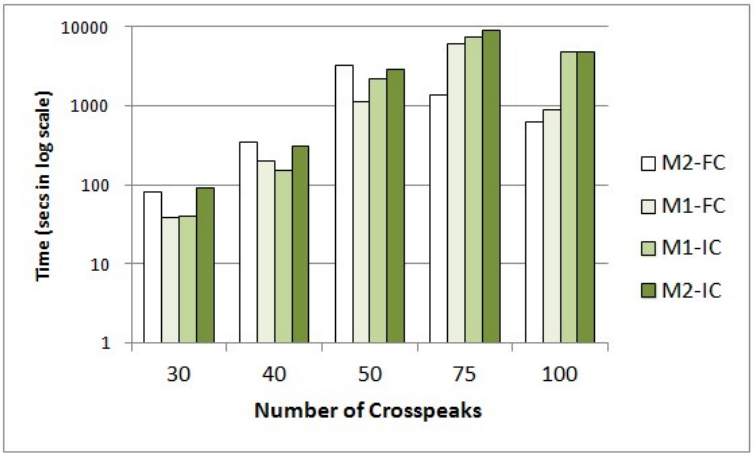


FIGURE 3.13: Sum of computing times for 35 instances solved within 1 hour.

The results indicate that the use of Branch & Cut approach to separate fractional cycles in the vertices of the branching tree is advisable. An additional confirmation of this statement can be deduced from Figure 3.13, where we plot the total time spent to solve the 50 instances by the 4 methods. Such indication would be biased by the fact that a different number of solutions may be obtained by various methods when the maximum computation time was reached. For this reason, in the chart we consider only the 35 problems for which all the methods have found the total number of optimal solutions within the time bound. Figure 3.13 highlights the importance of using the M1-FC and M2-FC models for saving CPU time. In particular, M2-FC was able to save in average 31% of computing time with respect to M1-FC, 61% w.r.t. M1-IC and 66% w.r.t. M2-FC.

Let us complete the analysis on the execution time by comparing the methods on a subset of 23 problems, for which all of them found the same set of solutions. In Table 3.13 we report the total solution times for each problems on this subset of instances; in the last column we compute the percentage of improvement in terms of time, i.e., the percentage of computational time saved by the fastest method over the second fastest in finding the solution. The best method in each row is highlighted wit a bold font. We can observe that the methods using FC performed better, and that the improvements obtained in solution time were quite large.

Prob. Idx	Exp. Type	# of Crosspeaks	method M1-IC	method M1-FC	method M2-IC	method M2-FC	Improve over second best
1	HCP	30	0.14	0.31	0.59	**0.08**	42.86%
2	HCP	30	2.56	**1.06**	1.81	1.37	22.63%
3	HCP	30	7.71	**3.2**	5.52	3.59	10.86%
4	HCP	30	8.25	**3.37**	10.7	3.71	9.16%
5	HCP	30	21.68	11.89	29.2	**9.34**	21.45%
6	HCP	40	2.5	**0.51**	1.78	1.84	71.35%
7	HCP	40	5.55	**2.73**	5.24	2.93	6.83%
8	HCP	40	36.02	6.61	12.65	**5.74**	13.16%
9	HCP	40	9.33	7.63	12.95	**6.24**	18.22%
10	HCP	40	29.47	**9.33**	23.65	13.1	28.78%
11	HCP	50	18.03	15.09	32.82	**8.08**	46.45%
12	HCP	50	85.18	65.69	50.7	**17.77**	64.95%
13	HCP	75	54.94	205.94	337.87	**35.01**	36.28%
14	HCP	75	53.82	227.28	265.01	**43.56**	19.06%
15	NOESY	30	0.64	**0.39**	0.58	0.59	32.76%
16	NOESY	30	1.9	5.87	12.78	**1.56**	17.89%
17	NOESY	30	6.24	**2.06**	3.73	2.61	21.07%
18	NOESY	40	4.49	**1.81**	2.7	2.95	32.96%
19	NOESY	40	47.66	**31.51**	62.68	34.3	8.13%
20	NOESY	40	68.45	42.88	95.99	**36.02**	16.00%
21	NOESY	50	220.3	163.58	839.33	**92.54**	43.43%
22	NOESY	50	**157.92**	743.91	531.54	209.1	24.48%
23	NOESY	50	513.8	649.92	593.12	**322.63**	37.21%

TABLE 3.13: Solution times (secs) for a subset of 23 problems where the methods where the performances can be compared straight-forwardly. Best performance for each row in **bold**

In order to analyze the efficiency of the methods from a biochemical point of view, we executed a set of tests taking into consideration the type of NMR map used. Let us recall that the number of colors composing the sequence of the path is three for a heteronuclear experiment and two for a homonuclear experiment (see Section 3.3.2).

	Experiment type	
Method	homonuclear	heteronuclear
M1-FC	10	5
M1-IC	1	1
M2-FC	4	9
M2-IC	0	0

TABLE 3.14: Method domination according to the experiment type (instances with size $\geq$ 50 cross-peaks).

In Table 3.14 we report the dominations related to the type of experiment in problems with 50, 75, and 100 cross-peaks. We can observe that for instances with smaller size the difference in methods performance was not meaningful. Moreover, M2-FC appeared superior for heteronuclear maps, while M1-FC performs better for homonuclear experiments.

In Figure 3.14 we can see how much M1-FC was faster in finding solutions for homonuclear experiments, and M2-FC for heteronuclear ones when large instances are considered. Let us remark that the barplot is referred to the total computing times spent by each model for solving all instances in Table 3.14.

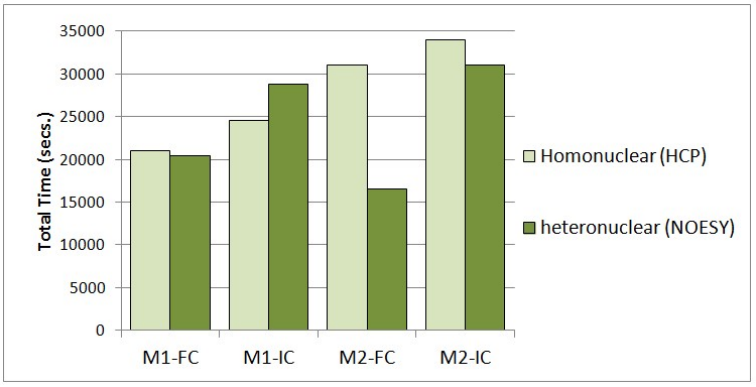


FIGURE 3.14: Computing time for instances with 50-100 cross-peaks.

Last step of our analysis was to compare both formulations in terms of number of cycles separated by each method. In this way we could to discover if the number of fractional cycles detected and cut by M1-FC and M2-FC was somehow related with the number of integral cycles cut by the relaxed formulations of M1-IC and M2-IC. Top panels of Figure 3.15 show that fractional separation methods (M1-FC and M2-FC) detected a higher number of cycles than integral ones (M1-IC and M2-IC). The bottom panels, instead, show that models with the same cycle separation method performed similarly, although it is more evident for M1-IC and M2-IC. In the other hand, from the comparison of M1-FC with M2-FC we note that the latter requires a larger number of cuts in about 66% of the considered problems, with a total of 25.483 cuts compared with 20.595 required by M1-FC.

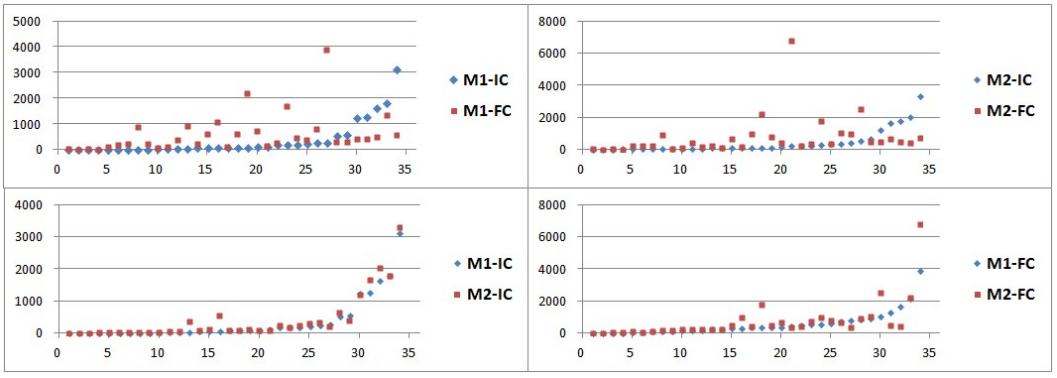


FIGURE 3.15: Number of cycles detected by each method for 35 instances solved within 1 hour: Model 1 with Integer and Fractional Cuts (top-left panel); Model 2 with Integer and Fractional Cuts (top-right panel); Model 1 and Model 2 with Fractional Cuts (bottom-right panel); Model 1 and Model 2 with Integer Cuts (bottom-left panel).

3.6.3 The litmus test

As above mentioned, not much experimental data of already solved cases has been collected and publicly available. Therefore, in order to check if one among our solutions

correctly reconstructed the assignment, besides the generated instance, we also used a subset of instances collected in [129] for which the original assignment pathway was a priori known. In Figure 3.16 there is the shortest 3D spectrum from this subset, and the original assignment pathway (the optimal solution) is drawn on the spectrum.

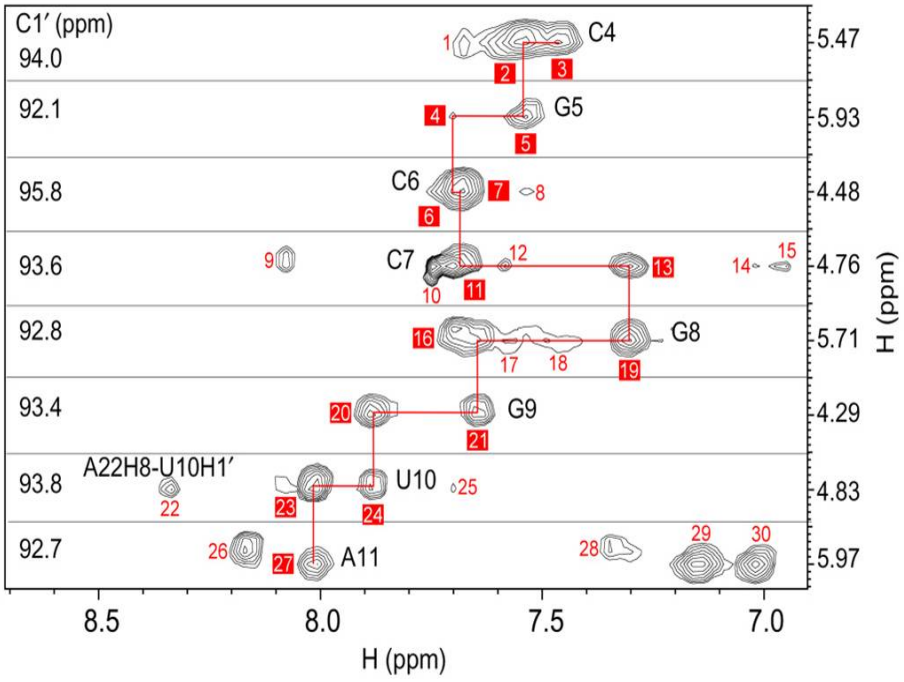


FIGURE 3.16: The original assignment pathway reconstructed for the heteronuclear sample r(CGCCGGUA) with 30 cross-peaks.

3.7 Conclusions

In this chapter we described the problem of assignment pathway reconstruction of a RNA molecule by 3D NMR maps which arises in the field of structural bioinformatics. This problem is the first computational step in elucidation of RNA tertiary structure [70, 129]. The idea is that determining the sequence of interactions among atoms involved in the NMR experiment can lead to determine the shape of such biological molecule. We proved that assignment pathway problem is NP-hard, and showed a formulation based on edge-colored graphs. Taking into account that interactions between consecutive nuclei in the NMR spectrum are different according to the type of residue along the RNA chain, each color in the graph represented a type of interaction. Thus, we represented the sequence of interactions as the problem of finding a longest (hamiltonian in the best case) path under the constraint that the edges of the path followed a given order of colors. We defined this problem as the Orderly Colored Longest Path problem on a c-edge-colored

graph (OCLP). We also showed how the OCLP problem can be adapted to model different types of real problems. Next, we described three alternative IP formulations used to solve the OCLP problem, which differ in the way the orderly colored paths over a directed network. In order to prove the efficacy of this approach, we tested our models over two sets of randomly generated problems with different characteristics.

The first set of experiments concerned instances generated as edge-colored graph on which paths of 2 or 3 colors were sought. The comparison among the three formulations in the first set of experiments (Section 3.6.1) has shown that two models, where cycle separation was performed iteratively adding constraints to a lighter formulation, performed much better. For this reason we have considered a second set of experiments focused only on these two models (Section 3.6.2). We developed a Branch & Cut procedure that uses a cycle separation polynomial oracle in order to enumerate all optimal solutions for the majority of the considered problems on a standard computer.

Since assignment of cross-peaks on the 3D NMR maps recorded for RNA molecules is a relatively new, not much expert knowledge concerning the problem, and not experimental data collected for already solved cases, are available for an extended analysis. This made us propose an instance generator that can be used to test automated solutions methods designed for the problem. Such generator can be of use in future research also to test alternative methods. The computational results of the second set of experiments have confirmed that the proposed models are valid options to solve 3D-APP problems of realistic sizes. In particular, most of large size instances (100 cross-peaks) have been solved within the time bound of one hour.

Future work will follow multiple directions: extending the test cases, including large real experimental data, developing of theoretical conjectures for the OCLP (i.e., about the presence of OCLP (or OCHP) in the graph related with some its characteristics), improving the formulations and the whole assignment procedure.

Chapter 4

The discretizable distance geometry problem

In this chapter, we introduce the Distance Geometry Problem (DGP), which is a well-known problem with applications in biology, statistics, and engineering. We focus on two particular applications of this problem, one in protein modeling and one in sensor networks localization. After giving a brief review of the existing approaches to the solution of this problem, we show few combinatorial requirements necessary to reduce the search space from continuous to discrete, defining the *discretizable* distance geometry problem. Finally, we describe the Branch & Prune (BP) algorithm, and two its versions designed to solve the DGP both in protein modeling and in sensor networks localization frameworks. BP is an exact and exhaustive combinatorial algorithm that examines all the valid embeddings of a given weighted graph $G = (V, E, d)$, under the hypothesis of existence of a given order on V. BP computes two possible positions for the current vertex, aimed to build a binary tree of solutions for the problem. At each step of the algorithm, if such positions satisfy some feasibility tests, they are actually added to the tree. We conclude the chapter showing some computational results related to these two BP versions to solve the two above mentioned DGP applications. This chapter is mainly based on the published papers [94, 97, 105] and the work presented in [51].

4.1 Introduction

The Distance Geometry Problem (DGP) consists of finding an embedding in $\mathbb{R}^k$ of a weighted undirected graph, where the edge weights are equal to the corresponding Euclidean distances in the embedding [25]. This problem has applications in many scientific and engineering fields, such as protein structure determination [26], graph drawing [73],

and wireless sensor networks localization [10], to name a few. In particular, protein structure determination is also well-known as the Molecular Distance Geometry Problem (MDGP), and it is the problem of determining the coordinates of the atoms in a molecule given the distances between certain pairs of atoms. Here a subset of interatomic distances may be known from the type of chemical bonds between atoms, or by means of Nuclear Magnetic Resonance (NMR) experiments. Typically, the MDGP involves placing atoms in $\mathbb{R}^3$ [74, 90, 145]. The Graph Drawing Problem is the problem of deriving representations in the plane or in the three-dimensional space of graphs, with the aim of finding visualizations of certain properties of the graph [32]. In the Sensor Networks Localization Problem[1] (SNLP) a pair of wireless sensors can estimate their distance by measuring the quantity of battery power necessary to a two-way communication. One particular property of the SNLP, which distinguishes it from MDGP, is that sensor network often has a subset of fixed sensors, called *anchors*, whose positions are known in advance. Such anchor positions can be used to get a subset of Euclidean distances which, in turn, can be used to determine the coordinates in $\mathbb{R}^2$ of the remaining sensors [9, 132]. In this chapter we mainly focused on the MDGP and on the SNLP.

Although the DGP is often formulated as a global optimization problem and solved by continuous methods, an embedding can be computed by a discrete search algorithm if the instances respect a few combinatorial requirements. This discretization process was previously developed in $\mathbb{R}^3$ basing on the observation that, under certain hypothesis, three spheres in $\mathbb{R}^3$ intersect in at most two points. A similar observation for the intersection of two circumferences in $\mathbb{R}^2$ leads to a polynomial time algorithm for the SNLP, as shown in [9]. Methods based on this kind of techniques are very common in many localization algorithms because they exploit the same principle used in GPS, known as hyperbolic trilateration, which allows to determine locations of points by measurement of distances, by using the geometry of circles, spheres, or triangles [109]. A survey on both continuous and discrete solution methods of these problems is given in [96].

The chapter is structured as follows. In Section 4.2 we introduce the mathematical notation and define the main problems in distance geometry, both for an arbitrary $k > 0$, as spatial dimension index, and for $k = 2, 3$. We conclude this section by examining the problem according to the density of the graph. In Section 4.3 we give a brief review of the existing continuous approaches to the solution of the DGP. In Section 4.4 we introduce the combinatorial property for the DGP and the corresponding combinatorial optimization problem. We also show an algorithm developed to solve the problem of

[1] A wireless sensor network usually consists of up to several hundred small autonomous devices to measure some physical parameters. Each device contains a processing unit, a radio transmitter and a receiver in order to be able to communicate with its neighbors.

finding an appropriate order of the vertex set in order to satisfy this combinatorial property. In Section 4.5 we explain a Branch & Prune algorithm for solving the DGP on a discrete search space in $\mathbb{R}^k$, both for $k > 0$ and for $k = 2, 3$. In Section 4.6 we discuss briefly both the problem and the algorithm complexity. In Section 4.7 we provide some experimental results on two sets of simulated instances. In Section 4.8 we draw some conclusions and describe some possible lines for future works.

4.2 Notations and Definitions

Given an undirected graph $G = (V, E)$where $|V| = n$ and $|E| = m$. We define:

1. $N(v) = \{u \in V | \{u, v\} \in E\}$ is the *neighborhood* of a given vertex $v \in V$.
2. $\gamma(v) = \{u \in V | u < v\}$ is the set of *predecessors* of v with respect to an order on V. Therefore we indicate with $N(v) \cap \gamma(v)$ the set of *adjacent predecessors* of a vertex $v \in V$.
3. $\rho(v) = |\gamma(v)| + 1$ is the *rank* of a vertex $v \in V$.

The embedding of a graph
Let $G = (V, E)$ be an undirected graph. Given an Euclidean space $\mathbb{R}^k$, an *embedding* of G in $\mathbb{R}^k$ is a function $x : G \to \mathbb{R}^k$ such that x maps V to a set of n points in $\mathbb{R}^k$ and E to a set of m line segments in $\mathbb{R}^k$.

According to this definition, a graph embedding is the representation of a graph in a Euclidean space in order to preserve its connectivity.

The Distance Geometry Problem
Let $G = (V, E, d)$ be an undirected weighted graph where a nonnegative function $d : E \to \mathbb{R}_+$ which assigns to each edge $\{u, v\} \in E$ a weight $d_{u,v}$. Given an integer $k > 0$. The DGP is the problem of finding an embedding x in $\mathbb{R}^k$ such that Euclidean distances between pairs of points are equal to the edge weights

$$\forall \{u, v\} \in E \quad ||x_u - x_v|| = d_{u,v}, \tag{4.1}$$

where $|| \cdot ||$ is the Euclidean norm computed between x_u and x_v, whereas $d_{u,v}$ refers to the known distances.

The DGP is a generalization of MDGP and SNLP to arbitrary dimension.

The Molecular Distance Geometry Problem
Let $G = (V, E, d)$ be an instance of the DGP in $\mathbb{R}^3$ such that each vertex $u \in V$ is an

atom. For each pair of vertices u and v in V, there exists an edge in E if the Euclidean distance $d_{u,v}$ is known either chemical bond analysis or NMR experiments. The MDGP consists in finding an embedding $x \in \mathbb{R}^3$ such that (4.1) holds.

The Sensor Networks Localization Problem
Let $G = (V, E, d)$ be an instance of the DGP in $\mathbb{R}^2$ such that each vertex $u \in V$ is a sensor. Given a radio range $R \in \mathbb{R}_+$, there exists in G an edge between two vertices u and v if and only if their Euclidean distance $d_{u,v}$ is known, and $||x_u - x_v|| \leq R$. Given a subset $U \subseteq V$ and an embedding $x' : U \to \mathbb{R}^2$ such that $||x'_u - x'_v|| = d_{u,v}$ for all $\{u, v\} \in E'$ in $G[U]$. The SNLP consists in finding an extension $x : V \to \mathbb{R}^2$ of x' which satisfies (4.1).

It is evident that SNLP $\supseteq$ MDGP, indeed, if $U = \emptyset$ then SNLP = MDGP. Therefore, a general method of solving the SNLP also solves the MDGP.

4.2.1 Graph distance density: complete vs sparse

The set E may not necessarily contain all possible $\{u, v\}$ pairs. We say that the problem has a sparse distance data, if E has only a subset of all $\{u, v\}$ pairs; otherwise, we say that it has a complete set of distances or dense distance data. The distances may not be provided as exact values and, in many cases, may be given in estimated ranges. When the exact distances are provided, the problem is formulated as (4.1), and it can then be solved in polynomial time. A solution with such a set of distance data can be obtained efficiently by using Singular Value Decomposition (SVD) of an induced distance matrix [52].
Let us assume that a set of coordinates $x_1, \cdots, x_n$ can be found for a given set of distances $d_{u,v}$, where $u, v = 1, \cdots, n$. Then, $||x_u - x_v|| = d_{u,v}$ for all $u, v = 1, \cdots, n$, and

$$||x_u||^2 - 2x_u^\top x_v + ||x_v||^2 = d_{u,v}^2, \qquad u, v = 1, \cdots, n. \tag{4.2}$$

Since the graph structure is invariant under any translation and rotation, we set a reference system so that the origin is located at the last vertex (e.g., in a three-dimensional space we set $x_n = (0, 0, 0)^\top$). It follows that

$$d_{u,n}^2 - 2x_u^\top x_v + d_{v,n}^2 = d_{u,v}^2, \qquad u, v = 1, \cdots, n-1. \tag{4.3}$$

Given a coordinate matrix X such that

$$X = \{x_{u,v} : \quad u = 1, \ldots, n-1, \quad v = 1, \ldots, k\} \tag{4.4}$$

and an introduced matrix D such that

$$D = \{(d_{u,n}^2 - d_{u,v}^2 + d_{v,n}^2)/2 : \quad u, v = 1, \ldots, n-1\}. \tag{4.5}$$

Then, $XX^\top = D$ and D must be of maximum rank k.

Theorem 4.1. *Let $\{d_{u,v} : u, v = 1, \ldots, n\}$ be a set of distances in $\mathbb{R}^k$, for some $k \leq n$. Then, the induced matrix D defined in (4.5) is of maximum rank k.*

Proof. It follows from the fact that $D = XX^\top$ for a coordinate matrix X in $\mathbb{R}^{n-1} \times \mathbb{R}^k$ and X is of maximum rank k. □

The equation $D = XX^\top$ can be solved in $O(kn^2)$ operations by using the SVD of D [68], and therefore, the DGP with a complete with a complete set of exact distances can be solved in polynomial time.
Note that, although in practice the distances may not be available for all the pairs of vertices, the solution of the problem with all exact distances can still be important for the solution of the general problem on a sparse set of distances. For example, if all the distances in a subset of vertices $S \subseteq V$ are known, but the position of these vertices does not, then we can determine such coordinates by solving a DGP with all exact distances for the subset S. The procedure may also be applied repeatedly until no such subsets of vertices can be found [122].

When the distances are inexact, or distance ranges or bounds, the solution is generally not unique, and there may exists a set of solutions that may all be of interest in practice. For example, in protein modeling the distances are often provided with some estimated bounds. The related DGP then becomes to find the coordinates $x_1, \cdots, x_n$ of the vertices, so that the distances $d_{u,v}$ between vertices v and v are within their estimated lower and upper bounds, $l_{u,v}$ and $u_{u,v}$, respectively, for all $\{u, v\}$ in a subset E of all pairs of vertices. That is,

$$l_{u,v} \leq ||x_u - x_v|| \leq u_{u,v} \quad \forall \{u, v\} \in E \tag{4.6}$$

Let $d_{u,v} = (l_{u,v} + u_{u,v})/2$ and $\varepsilon = (u_{u,v} - l_{u,v})/2$. We can rewrite the problem (4.6) as

$$|||x_u - x_v|| - d_{u,v}| \leq \varepsilon \quad \forall \{u, v\} \in E \tag{4.7}$$

Then, the problem can be viewed as to find an approximate solution to the DGP for a set of exact distances $d_{u,v}$ with each distance $||x_u - x_v||$ allowed to have an error ε from $d_{u,v}$. Such a solution is called an ε-approximation solution. If large errors are allowed, an approximate solution is certainly easier to obtain than an exact solution.
The method that we shall discuss in this chapter concerns only instances with exact

distance data, although a version of BP which works with interval of distance ranges from NMR data has already been implemented [91]. However, the BP version for not exact distance on SNLP instances is under completion.

4.3 A continuous approach for solving the DGP

As mentioned earlier in Section 4.2, DGPs are often formulated and solved by continuous method. The system of nonlinear equations (4.1) can be re-cast as a penalty function to be minimized:

$$\min_{x} \sum_{\{u,v\}\in E} (||x_u - x_v||^2 - d_{u,v}^2)^2. \tag{4.8}$$

Although this function is a sum of squares, this is a nonconvex optimization problem in x, and falls into the category of global optimization admitting many local minima [96]. It is clear that x is a solution for (4.8) if and only if the value of the penalty function in the feasible solution set X is exactly $\emptyset$.

An approach to the solution of the DGP that replaces the large optimization problem in (4.8) by a sequence of smaller ones has been proposed in [76]. The author showed that the structure can be exploited by using a divide-and-conquer algorithm, which helps reduce the complexity of the problem. Moreover, he developed a software package called ABBIE [76] for the determination of molecular structure with a given set of distances. The program first decomposes the graph recursively into subgraphs with unique 3D embeddings. The smaller embedding problems are then solved by minimizing the least-square error function (4.8). Another well-known approach to the MDGP is DGSOL, a method in which the penalty function is approximated by a sequence of smoother functions converging to the original one [103]. In detail, the global smoothing method first transforms a weighted function similar to (4.8) into a set of smoother functions with fewer local minima. The method uses these local minima to trace their changes when the smoother functions are changed back to the original one. In this way a global minimum is located at the end. DGSOL has been efficiently applied to some small to medium-sized test problems with around 200 atoms. Although the method is still fast even on large instances, the solution quality decreases with the growth of the graph dimension. In [47] we can find the embedding algorithm, one of the major contribution to the development of distance geometry in protein modeling, where there is the first application of NMR spectroscopy experiments as distance data. This method determines the coordinates of the atoms for given set of interatomic distances (or their ranges) by exploiting some geometric properties, like triangle inequality, to estimate the missing distance. The distance geometry literature also includes a class of semidefinite

programming (SDP) based methods, which are often used to solve SNLP instances [22, 37]. These methods relax the SNLP to a weighted semidefinite programming problem using the linear mapping between Euclidean distance matrices and semidefinite matrices. Then the SDP problem is solved by primal-dual interior point solvers. We conclude this brief, and not complete, overview on continuous method with the *Geometric build-up* algorithm originally described in [52], and recently revisited in [123, 141]. This algorithm works directly on the given distances by exploiting the special structure of a given problem, and hence may be able to solve the problem more efficiently than a general approach, such as we explain more deeply in Section 4.3.1.

4.3.1 The geometric build-up algorithm

The geometric build-up algorithm solves the MDGP on sparse graphs in $\mathbb{R}^3$ [52]. This algorithm is based on geometric relationship between coordinates and distances to the atoms of a molecule. Central to the algorithm is the idea to determine only a small group of atoms at the beginning, and then complete the whole molecule by repeatedly determining one or more atoms every time using the available distances between the determined and undetermined atoms. I.e., let us assume that it is possible to determine the coordinates of at least four atoms, which are marked as fixed; the remaining ones are non-fixed. Then the coordinates of a non-fixed atom a can be calculated by using the coordinates of four non-coplanar fixed atoms such that the distances between any of these four atoms and the atom a are known. If such four atoms are found, the atom a changes its status and it becomes fixed.

We consider that an atom a is a vertex $v \in V$ in the graph. Let u_1, u_2, u_3, u_4 be four vertices representing the four fixed atoms whose Cartesian coordinates are already known. Let us suppose that the Euclidean distances among vertex v and the vertices u_1, u_2, u_3, u_4 namely d_{v,u_i}, for $i \in \{1, 2, 3, 4\}$, are known. Then $x_v = x(v)$ is a solution of the following system of equations:

$$
\begin{aligned}
||x_v - x_{u_1}|| &= d_{v,u_1}, \\
||x_v - x_{u_2}|| &= d_{v,u_2}, \\
||x_v - x_{u_3}|| &= d_{v,u_3}, \\
||x_v - x_{u_4}|| &= d_{v,u_4}.
\end{aligned}
$$

Squaring both sides of these equations, we have:

$$\begin{aligned}
||x_v||^2 - 2x_v^\top x_{u_1} + ||x_{u_1}||^2 &= d_{v,u_1}^2, \\
||x_v||^2 - 2x_v^\top x_{u_2} + ||x_{u_2}||^2 &= d_{v,u_2}^2, \\
||x_v||^2 - 2x_v^\top x_{u_3} + ||x_{u_3}||^2 &= d_{v,u_3}^2, \\
||x_v||^2 - 2x_v^\top x_{u_4} + ||x_{u_4}||^2 &= d_{v,u_4}^2.
\end{aligned}$$

By subtracting one of the above equations from the others, we have a linear system that admits a unique solution of x_v since u_1, u_2, u_3, u_4 are non-coplanar. For example, if we subtract equation i from equation $i+1$ for $i = 1, 2, 3$, then the quadratic terms for x_v can be eliminated, and we obtain

$$\begin{aligned}
-2(x_{u_2} - x_{u_1})^\top x_v &= (d_{v,u_2}^2 - d_{v,u_1}^2) - (||x_{u_2}||^2 - ||x_{u_1}||^2) \\
-2(x_{u_3} - x_{u_2})^\top x_v &= (d_{v,u_3}^2 - d_{v,u_2}^2) - (||x_{u_3}||^2 - ||x_{u_2}||^2) \\
-2(x_{u_4} - x_{u_3})^\top x_v &= (d_{v,u_4}^2 - d_{v,u_3}^2) - (||x_{u_4}||^2 - ||x_{u_3}||^2).
\end{aligned}$$

Let A be a matrix and b a vector such that

$$A = -2 \begin{bmatrix} (x_{u_2} - x_{u_1})^\top \\ (x_{u_3} - x_{u_2})^\top \\ (x_{u_4} - x_{u_3})^\top \end{bmatrix}, \quad b = \begin{bmatrix} (d_{v,u_2}^2 - d_{v,u_1}^2) - (||x_{u_2}||^2 - ||x_{u_1}||^2) \\ (d_{v,u_3}^2 - d_{v,u_2}^2) - (||x_{u_3}||^2 - ||x_{u_2}||^2) \\ (d_{v,u_4}^2 - d_{v,u_3}^2) - (||x_{u_4}||^2 - ||x_{u_3}||^2) \end{bmatrix}.$$

Since u_1, u_2, u_3, u_4 are not in the same plane, A must be nonsingular, and we can solve the linear system $Ax_v = b$ in order to obtain a unique solution for x_v. In this case, all distances are known, then solving the system requires only constant time [52].

In [52] the authors report that the GB algorithm is very sensitive to the numerical errors introduced in computing the atomic coordinates. These numerical errors are controlled in the updated version proposed in [141]; whereas in [123] are described the extension of the algorithm to handling inexact distance data. GB solves the MDGP in linear time if G is a complete graph and $d_{u,v} \in \mathbb{Q}_+$ for all $\{u, v\} \in E$.

Requiring the knowledge of the distances of four previously embedded adjacent vertices limits the extention of the algorithm to instances with relatively dense graphs. In fact, distances are usually hard to obtain (this is true especially for sensor networks). An effort should be made in order to weaken this requirement. Although similar concepts were already known in rigidity [76], the first work providing an iterative discrete search algorithm for the MDGP that only requires three (rather than four) previously embedded adjacent vertices is BP [94], a mixed combinatorial algorithm used in the development of the MD-jeep software [107] to solve problems related to protein molecules (see Section 4.5 for more details). The main difference between GB and the BP algorithm solving the DGP is that BP exploits a given order on V. This restriction allows BP to be more

reliable, efficient and complete than continuous methods. In particular, it has been shown that BP works well in proteins structure determination [94, 96, 97, 105–107].

The original contribution presented in this dissertation is the development of a version of BP able to solve also SNLP instances [51]. In the rest of this chapter we describe the discretization principle based of BP, and we highlight the differences between its two versions.

4.4 Discretizing the search space

Although the DGP implicitly requires a search in continuous space, if an appropriate order is given on V, we can show that the search space has a finite number of valid embeddings, up to translations and rotations. In such framework it is located the Discretizable Distance Geometry Problem (DDGP), which consists of a subclass of instances for the DGP with a combinatorial property in order to be solved by a discrete search algorithm.

The Discretizable Distance Geometry Problem
Given an instance of the DGP $G = (V, E, d)$, an integer $k > 0$ and an order on V such that are verified the following assumptions:

1. the first $k + 1$ vertices in V compose a clique;
2. for each $v > k$ there exists a subset $U_v \subseteq N(v) \cap \gamma(v)$ of at least k elements such that:
   a) $G[U_v]$ is a k-clique in G;
   b) strict simplex inequalities $\Delta_k(U_v, d) > 0$ hold,

find an embedding x in $\mathbb{R}^k$ such that (4.1) holds.

These assumptions allow us to extend the partial embedding x' composed by the first k vertices. Here, the $\Delta_k(U_v, d)$ is the content of the k-simplex defined by the k-clique $G[U_v]$ and the vertex v:

$$\Delta_k(U_v, d) = \sqrt{\frac{(-1)^{k+1}}{2^k(k!)^2}|CM(U_v)|} \tag{4.9}$$

computed by using the Cayley-Menger determinant [25]:

$$CM(U_v) = \begin{bmatrix} 0 & 1 & 1 & \dots & 1 \\ 1 & 0 & d_{1,2}^2 & \dots & d_{1,k+1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & d_{1,k+1}^2 & d_{2,k+1}^2 & \dots & 0 \end{bmatrix} \tag{4.10}$$

with $d_{ij} = ||x'_i - x'_j||$ for all $i, j \in \{1, \cdots, k+1\}$.

The strict simplex inequalities and the k-clique requirement guarantee that the k-simplex has nonzero volume. Note that only the distances of the simplex edges are necessary to compute $\Delta_k(U_v, d)$, rather that the coordinates of the points in U_v; the required information can be encoded as a complete graph on $k+1$ vertices with the distances as edge weights.
When $k = 3$ the DDGP can be reduce to the Discretizable Molecular Distance Geometry Problem (DMDGP) proposed in [94, 97], where the strict simplex inequalities are strict triangular inequalities [89]. In fact, in a three-dimensional space the 3-simplex is a pyramid whose base is the triangle formed by the 3-clique $G[U_v]$, and the apex is the vertex v. Therefore, the distances are the edge lengths of the pyramid. The volume of this simplex is zero when the points of the clique $G[U_v]$ are collinear (the base is a segment).

The Discretizable Molecular Distance Geometry Problem
Given an instance of the MDGP $G = (V, E, d)$ and an order on V such that are verified the following assumptions:

1. the first 4 vertices in V compose a clique;
2. for each $v > 3$ there exists a subset $U_v \subseteq N(v) \cap \gamma(v)$ containing its 3 immediate predecessors $\{v-1, v-2, v-3\}$ such that:
   a) $G[U_v]$ is a clique in G;
   b) strict triangular inequality $d_{v-3,v-1} < d_{v-3,v-2} + d_{v-2,v-1}$ holds,

find an embedding x in $\mathbb{R}^3$ such that (4.1) holds.

The set of adjacent predecessors $N(v) \cap \gamma(v)$ for an instance of the DMDGP has to contain the three *immediate* predecessors of v, instead of any triplet of its predecessors, as is required for a DDGP instance. Although this requirement is stronger than the one for the DDGP, it is very realistic for proteins molecules which have a natural ordered structure due to the particular positions of the atoms. This structure, indeed, determines particular geometric proprieties useful for a more efficient reordering of the atoms (see

Section 4.4.3). The idea is that along the protein backbone, atoms that are close in sequence are also close in distance, and consequently, they are also close in the three-dimensional conformation of the protein.

Assuming that all vertices preceding v are already embedded, we can calculate all mutual distances needed for the computation of $\Delta_3(U_v, d)$. If the strict triangular inequality holds, then the intersection can only have either one or two points, depending on whether the discriminant of a certain quadratic polynomial in x_v is zero or nonzero [45, 111]. We shall discuss this *finite sphere intersection* property in Section 4.4.1.

The DDGP may also represent a discretization of the SNLP. For the sake of clarity we refer to it as the Discretizable Sensor Networks Localization Problem (DSNLP).

The Discretizable Sensor Networks Localization Problem
Given an instance of the SNLP $G = (V, E, d)$ and an order on V such that are verified the following assumptions:

1. the first three vertices induce a clique in G;
2. for each $v > 2$ the set $|N(v) \cap \gamma(v)|$ contains at least two (different) elements.

find an embedding x in $\mathbb{R}^2$ such that (4.1) holds.

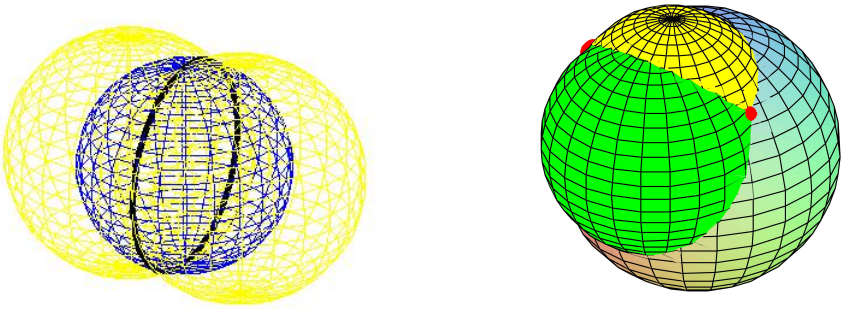
Working on a two-dimensional space the DSNLP assumptions can be expressed in a weaker form: it is enough that any sensor v interacts with at least other two sensors previously visited, whose distances from v are known. The 2-simplex is a triangle whose vertices are the current vertex v and a pair of adjacent predecessors of v. Therefore, the requirement of strict simplex inequalities becomes unnecessary because the computed Cayley-Menger determinant is always greater than zero. Moreover, the subgraph induced by the anchors can be used to build the initial embedding x', thus the set of anchors must have at least three elements. BP determines all possible embeddings of a DSNLP instance by extending x'.

4.4.1 Sphere intersections

As above mentioned, once the vertices of U_v have been embedded in $\mathbb{R}^k$, the known distances from vertices in U_v to a given v will enforce the position of v as the intersection of k spheres.
Let $S(x, r)$ be a sphere in $\mathbb{R}^k$ with center $x \in \mathbb{R}^k$ and radius $r \in \mathbb{R}_+$, and let

$$I = \bigcap_{v=1}^{k} S(x_v, d_{v,k+1})$$

be the intersection of k spheres with centers $x_1, \ldots, x_k$ and radii $d_{1,k+1}, \cdots, d_{k,k+1}$. The intersection of these k spheres in $\mathbb{R}^k$ might contain zero, one, two or uncountable points depending on the position of the centers and the lengths of the radii. In particular, if the k sphere centers are collinear in $\mathbb{R}^k$, for $k > 2$, then the set of distance matrices yield a Cayley-Menger determinant having value exactly zero (the simplex inequalities 2.b fail to hold), and their intersection I has uncountable cardinality (see, for example, the thick circle in Figure 4.1(a) in $\mathbb{R}^3$); when $k = 2$ it is provided if the two circumferences are centered in the same point, and consequently, they have equal radius lengths. On the other hand, if the k-clique 2.a cannot be realized, then $I = \emptyset$. In general, the strict



(a) sphere centers collinear

(b) sphere centers non-collinear

FIGURE 4.1: The intersection of three spheres in $\mathbb{R}^3$.

simplex inequalities 2.b and the k-clique requirement 2.a preserve the dimensionality of the k-simplex, guaranteing the finite sphere intersection property so that the intersection I can only contains:

- one point if $|U_v| = k + 1$ in $\mathbb{R}^k$. For instance, let us suppose that in $\mathbb{R}^2$ the vertex v has three adjacent predecessors v_1, v_2, v_3. The intersection of the three circumferences centered in v_1, v_2, v_3 with radii $d_{v_1,v}, d_{v_2,v}, d_{v_3,v}$ is just the single point v, which is a vertex of the polytope v, v_1, v_2, v_3.
- two points if $|U_v| = k$ in $\mathbb{R}^k$ (as showed in Figure 4.1(b) for $\mathbb{R}^3$).

Last two considerations have been formalized in [97] by few lemmata related to rigid graphs [57], which we introduce in the following. We assume that the probability of any point in $\mathbb{R}^k$ belonging to any given subset of $\mathbb{R}^k$ having Lebesgue measure zero is equal to zero. Based on this assumption, when we state "$\forall p \in P$, $F(p)$ has probability 1" for a certain well-formed formula F means that the statement $F(p)$ holds over a subset of P that has Lebesgue measure 1. Usually, this occurs whenever p is a geometrical statement about Euclidean space that fails to hold for strictly lower dimensional manifold, such as

a simplex in $\mathbb{R}^k$ composed by fewer than $k+1$ vertices. In this situation the collinearity causes an uncountable P as in Figure 4.1(a), and the Lesbegue measure is zero within the set of all possible (real) distance matrices. Let $G^v = G[\gamma(v) \cup \{v\}]$ be a valid embedding of $G[\gamma(v)]$.

Lemma 4.1. *If $|N(v) \cap \gamma(v)| = k$ then there are at most two distinct extensions of x valid for G^v. If one valid extension exists, then with probability 1, there are exactly two distinct valid extensions.*

Lemma 4.2. *If $|N(v) \cap \gamma(v)| > k$ then, with probability 1, there is at most one extension of x.*

Lemma 4.3. *With the notation of Lemma 4.1, if x' is a valid embedding for $G[U_v]$, then $\bar{z}$ is a reflection of z with respect to the hyperplane through the k points of x'.*

Proof. All proof of lemmata 4.1-4.3 are provided in [97]. □

4.4.2 The influence of rigidity

Since the graph may be sparse, the embedding may not be unique. There may be more than one way to position the points, and all the distance constraints can still be satisfied. If some of the points can be moved continuously without violating any distance constraints, the graph is called *flexible*; otherwise, it is called a *rigid* graph. Note that flexibility of a graph leads to infinitely many solutions to the DGP [76].
Rigidity and uniqueness of the distance graph can be important for the study of DGPs. In order for a graph to have a unique embedding, it is obvious that it must first be rigid. However, a rigid graph may still have multiple embeddings, for example, when it has partial reflections. Thus, another necessary condition for unique embeddability is that the graph does not have partial reflections. In a k-dimensional space this is guaranteed only if the graph is (k+1)-connected. All these properties can be used to exploit the structure of large graphs to find subgraphs that have unique embeddings. The embedding problem for a given distance graph can then be solved by dividing the graph into such subgraphs. The solutions found for any subgraph can finally be combined into a unique solution for the whole graph [76].

Reflections with respect to hyperplanes are isometries[2], and can therefore be represented by linear operators. If $a \in \mathbb{R}^k$ is the unit normal vector to a hyperplane H containing the origin, then the reflection operator R_0 w.r.t. H can be expressed in function of the

[2] An isometry is a transformation which maps elements from a metric space A to another metric space B such that the distances between the elements in B are equal to the distances between the elements in A.

standard basis by the matrix $I - 2aa^\top$, where I is the $K \times K$ identity matrix [31]. Let H be a hyperplane with equation $a^\top x = a_0$ (with $a_0 \neq 0$) and a_i be, for some $1 \leq i \leq k$, the nonzero coefficient of the smallest index in a. Then, the reflection operator R acting on a point $p \in \mathbb{R}^k$ w.r.t. H is given by

$$R(p) = R_0(p - \frac{a_0}{a_i}e_i) + \frac{a_0}{a_i}e_i,$$

where $e_i \in \mathbb{R}^k$ is the unit vector with 1 at index i and 0 elsewhere: we first translate p so that we can reflect it using R_0 w.r.t. the translation H containing the origin, then we perform the inverse translation of the reflection.

4.4.3 The importance of being ordered

It is clear that discretization requirements of the DDGP strongly depend from the ordering of the vertex set V. Since DDGP instances may not satisfy such requirements in the reality, we consider the problem of finding a good order (or determining that one such order does not exists) as a pre-processing step before to solve the DDGP [89].

The Discretization Vertex Order Problem (DVOP)
Given a simple undirected graph $G = (V, E)$ and a positive integer $k > 0$, establish whether there is an order on V such that:

1. the first k vertices induce a clique in G;
2. each $v \in V$ with rank $\rho(v) > k$ has $|N(v) \cap \gamma(v)| \geq k$.

Note that DVOP does not verify whether the order satisfies the strict simplex inequalities requirements. DVOP only allows us to embed the first k vertices uniquely, and it ensures us that any vertex in V has at least k adjacent predecessors. In fact, if a vertex v has fewer than k adjacent predecessors, then there may be infinitely many placements for it, which means that $|N(v) \cap \gamma(v)|$ does not allow to define an appropriate order on V.

An exponential time solution algorithm for this problem consists in finding a k-clique C in G, and to consider their vertices as first vertices of the new ordering. Then, all other vertices are greedily positioned in the new ordering by choosing any time the one with the largest number of adjacent predecessors, stopping whether this is smaller than k. A sketch of this reordering algorithm is given below. If k is a fixed constant, then this becomes a polynomial algorithm. Since DGP applications rarely require a variable k, this is a comforting result.

A reordering algorithm

Let $B = \emptyset$ be the set of the reordered vertices in G.
reorder(G)
while (a valid ordering is not found) **do**
  find a k-clique C in G;
  place the vertices of C at the beginning of new order: $B = C$;
  while $(V \setminus B \neq \emptyset)$ **do**
    find the vertex v in $V \setminus B$ with the largest number l of adjacent vertices in B;
    if $(l < k)$ **then**
      end while: there are no possible orderings for this choice of C;
    end if
    $B = B \cup \{v\}$;
  end while
end while

4.5 A Branch & Prune algorithm for solving the DDGP

In this Section we show a Branch & Prune (BP) algorithm originally drawn to solve MDGP instances [94, 97], and adapted in a later stage to solve also SNLP instances [51]. BP is an exact and exhaustive combinatorial algorithm that examines all the valid embeddings of a given weighted graph $G = (V, E, d)$, under the hypothesis of existence of a given order on V.

For solving the DDGP in a k-dimensional embedding space, the BP algorithm requires five arguments recursively:

1. the weighted simple undirected graph $G = (V, E, d)$;
2. a current vertex v to embed;
3. a subset $U_v \subseteq N(v)$ of k elements;
4. a valid embedding x' of a subgraph of G containing $G[U_v]$;
5. the set X of valid embeddings of G currently found.

The BP algorithm computes two possible positions for the current vertex v, with the intent of building a binary tree of solutions for the problem. The nodes of the tree at level v represent possible spatial positions i for the vertex v. In Figure 4.2, for example, we can see a binary tree T for $n = 6$. The blue boxes show a complete path

on T from the root x_1 to one its leaf node x_6. The path corresponds to the solution $(x_4, x_5, x_6) = (0, 1, 1)$ found starting from the initial clique (x_1, x_2, x_3). When a position fails some feasibility tests, the branch i is pruned, where the ith variable indicates which of the two possible choices for the vertex x_i have been taken. A more detailed description of pruning techniques is given in Section 4.5.3, whereas a sketch of BP is provided below.

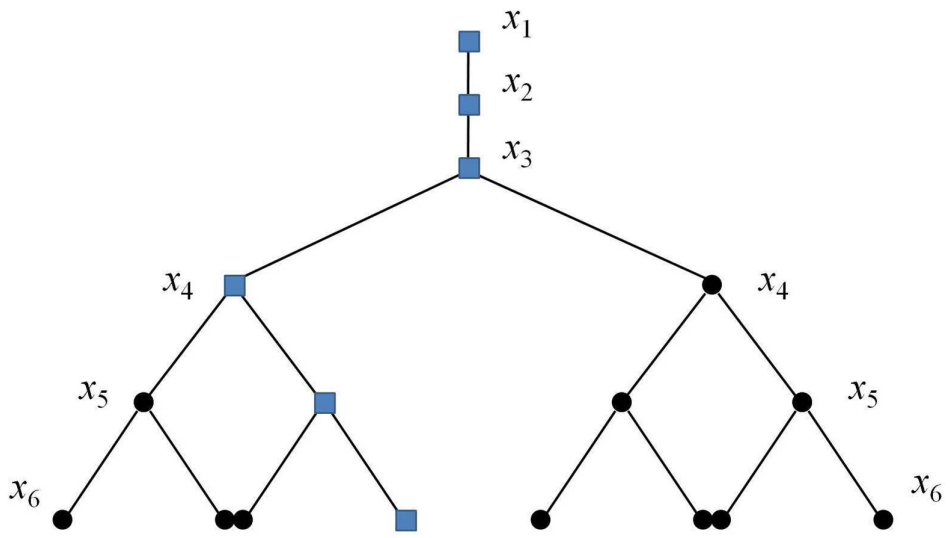


FIGURE 4.2: The binary tree T for $n = 6$ corresponding to the solution $(x_4, x_5, x_6) = (0, 1, 1)$.

Theorem 4.2. *At termination of BP, X contains all valid embeddings of G extending x'.*

For each leaf node of the tree generated by BP, the unique path to the root node encodes an embedding of G. By the Theorem 4.2, for which a proof is provided in [94], the paths from each leaf node at level $|V|$ encode all valid embeddings of G exending x'. We remark that the BP can be stopped after the first valid embedding has been found when just one solution of the DDGP is needed. It can also be allowed to proceed until all valid embeddings have been identified. This makes the BP algorithm complete. Since the first k nodes can be fixed, in the worst case the binary tree contains 2^{n-k} nodes. This makes BP a worst-case exponential algorithm.

BP algoritm

Let I be the intersection of the k spheres $S(x'_u, d_{u,v})$ for $u \in U_v$.
BP(G, v, U_v, x', X)
for $(i \in I)$ **do**
  compute the i^{th} position for the current vertex v: x_{v^i};
  check the feasibility of x_{v^i}: is $x = (x', x_{v^i})$ a valid embedding of $G[\{1, \ldots, v\}]$?
  if (the position x_{v^i} is feasible) **then**
    if $(v = n)$ **then**
      one solution is found: $x \rightarrow X$;
    else
      let $U'_v = (U_v \setminus \{\text{minU}_\text{v}\}) \cup \{\text{v}\}$;
      BP$(G, v + 1, U'_v, x, X)$;
    end if
  else
    the current branch is pruned;
  end if
end for

4.5.1 BP for the DMDGP

Given a DMDGP instance as above described, BP computes two new positions for any vertex v exploiting 3 immediate predecessors $\{v-1, v-2, v-3\}$ of v w.r.t. the vertex order imposed on V (before to run BP). The recursive algorithm starts with the call BP$(G, 4, U_v = \{1, 2, 3\}, x', \emptyset)$ where x' is the valid embedding of the first three vertices inducing a clique in G. This is possible only if all the assumptions described in Section 4.4 are satisfied. Under such conditions the sphere intersection property in Section 4.4.1 guarantees that the intersection I of three sphere centered at x_{v-1}, x_{v-2} and x_{v-3}, with radii $d_{v-1,v}$, $d_{v-2,v}$, and $d_{v-3,v}$, consists of at most two distinct points. In particular, $x_{v^1} \in I$ is the reflection of $x_{v^0} \in I$ through the hyperplane defined by x_{v-1}, x_{v-2}, and x_{v-3}.

In the case of DMDGP, the problem of intersecting the three spheres can be replaced by the problem of finding the possible torsion angles along a backbone of atoms of the molecule. It has be proved that there are only two possible torsion angles for each quadruplet of consecutive atoms $\{v-3, v-2, v-1, v\}$. These two torsion angles correspond to two possible positions for the last atom of the quadruplet [95]. Below we describe how the problem of intersecting the three spheres can be replaced by the

problem of finding the possible torsion angles along the protein backbone related to the order on V by using consecutive atoms.

Without loss of generality, fixed three consecutive atoms $\{v-3, v-2, v-1\}$ in the sequence, we can express the cosine of the torsion angle ω_v in terms of the distances $r_{v-2}, d_{v-2,v}, d_{v-3,v}$ and the bond angle θ_{v-1}, θ_v as follows:

$$\cos\omega_v = \frac{r_{v-2}^2 + d_{v-2,v}^2 - 2r_{v-2}d_{v-2,v}\cos\theta_{v-1}\cos\theta_v - d_{v-3,v}^2}{2r_{v-2}d_{v-2,v}\sin\theta_{v-1}\sin\theta_v},$$

where r_v is the Euclidean distance (the bond length) between the atoms $v-1, v$, for all $v \in \{2, \ldots, n\}$; $\theta_v \in [0, \pi]$ is the *bond angle* between the segments joining the atoms $v-2, v-1$ and $v-1, v$, for all $v \in \{3, \ldots, n\}$; and $\omega_v \in [0, 2\pi]$ is the *torsion angle* between the normals through the planes defined by the atoms $v-3, v-2, v-1$ and $v-2, v-1, v$, for all $v \in \{4, \ldots, n\}$ (see Figure 4.3).

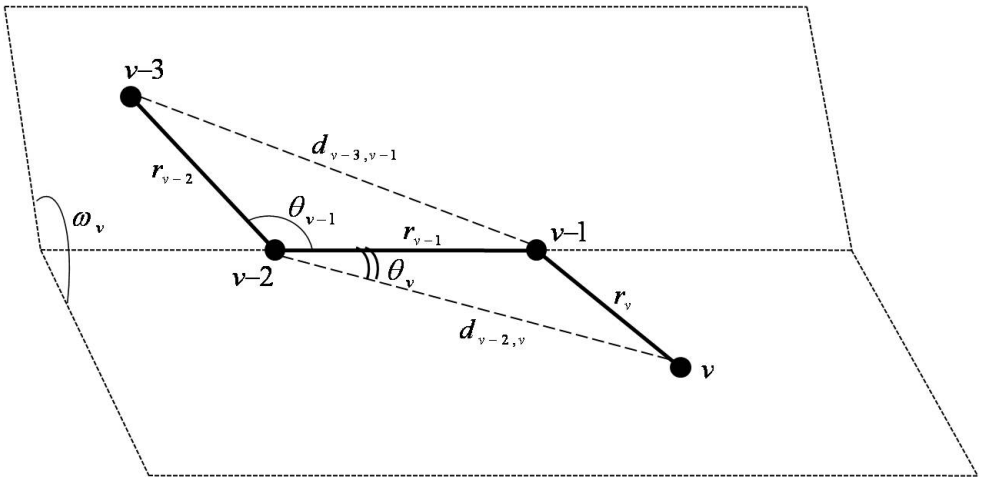


FIGURE 4.3: Definitions of bond lengths, bond angles and torsion angles.

Hence, if we know all the bond lengths (r_v), bond angles (θ_v), and distances between the atoms separated by three covalent bonds ($d_{v-3,v}$), we can calculate the cosine of the torsion angles defined by the atoms $\{v-3, v-2, v-1, v\}$ for $v = 3, \ldots, n$. This, in turn, can be used to compute the two possible positions x_{v^0}, x_{v^1} for the fourth atom v by determining

$$x_{v^0} = \begin{bmatrix} -r_{v-2} + r_{v-1}\cos\theta_{v-1} - r_v\cos\theta_{v-1}\cos\theta_v + r_v\sin\theta_{v-1}\sin\theta_v\cos\omega_v \\ r_{v-1}\sin\theta_{v-1} - r_v\sin\theta_{v-1}\cos\theta_v - r_v\cos\theta_{v-1}\sin\theta_v\cos\omega_v \\ -r_v\sin\theta_v\sqrt{1-(\cos\omega_v)^2} \end{bmatrix},$$

$$x_{v^1} = \begin{bmatrix} -r_{v-2} + r_{v-1}\cos\theta_{v-1} - r_v\cos\theta_{v-1}\cos\theta_v + r_v\sin\theta_{v-1}\sin\theta_v\cos\omega_v \\ r_{v-1}\sin\theta_{v-1} - r_v\sin\theta_{v-1}\cos\theta_v - r_v\cos\theta_{v-1}\sin\theta_v\cos\omega_v \\ +r_v\sin\theta_v\sqrt{1-(\cos\omega_v)^2} \end{bmatrix}.$$

For the fifth atom, we will obtain four possible positions, one for each combination of $\pm\sqrt{1-(\cos\omega_4)^2}$ and $\pm\sqrt{1-(\cos\omega_5)^2}$. As consequence, we can see that for the i-th atom we obtain 2^{i-3} possible positions (as it is shown in Figure 4.2). So, for a molecule shaped as a sequence of n atoms, we get 2^{n-3} possible sequences of torsion angles $\omega_4, \omega_5, \ldots, \omega_n$ each defining a different three-dimensional structure. By using the torsion matrices B_v, defined as

$$B_v = \begin{bmatrix} -\cos\theta_v & -\sin\theta_v & 0 & -r_v\cos\theta_v \\ \sin\theta_v\cos\omega_v & -\cos\theta_v\cos\omega_v & -\sin\omega_v & r_v\sin\theta_v\cos\omega_v \\ \sin\theta_v\sin\omega_v & -\cos\theta_v\sin\omega_v & \cos\omega_v & r_v\sin\theta_v\sin\omega_v \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

for $v = 1, \ldots, n$, we can convert a sequence of torsion angles into Cartesian coordinates x_1, x_2, x_3 in $\mathbb{R}^3$. In fact, at any iteration, BP computes B_v and solves the two following systems:

$$(x_{v^0}, 1)^\top = B_1 B_2 \ldots B_{v^0}(0,0,0,1)^\top$$
$$(x_{v^1}, 1)^\top = B_1 B_2 \ldots B_{v^1}(0,0,0,1)^\top.$$

4.5.2 BP for the DSNLP

We developed a new version of BP, which solves DSNLP instances by intersecting two circumferences in $\mathbb{R}^2$ [51].

Let $u, u' \in N(v) \cap \gamma(v)$ be a pair of precedents of $v \in V$. Then, $x_{v^1} \in I$ is the reflection of $x_{v^0} \in I$ through the straight line defined by x_u, $x_{u'}$. Thus, at each iteration, BP solves twice the following quadratic system:

$$\begin{cases} ||x_u - x_{v^i}|| = d_{u,v^i} \\ ||x_{u'} - x_{v^i}|| = d_{u',v^i}, \end{cases} \tag{4.11}$$

for both vertices $v^i = \{0, 1\}$. In literature there exist several method finding solutions of the system (4.11), such as the Gaussian Elimination or the Orthogonal Decomposition [45]. Let us remark that, whatever method is used, it is very important that the found solutions are very accurate. Indeed, they represent the possible positions for the graph vertices, which have to pass some tests for feasibility before being inserted in the binary tree. Therefore, if the found solutions for (4.11) are not accurate enough, then the pruning tests might reject them all, and no solutions are found.

While, in the case of the DMDGP, there exists the choice of considering torsion angles instead to compute the sphere intersections, this is not possible when we consider DSNLP instances. A sequence of quadratic systems need instead to be computed and, at each interaction of BP, we must be aware that some errors may be introduced in the computed coordinates. In order to keep the propagation of these errors as low as possible, we implemented the following strategy: at each iteration of BP, we try all the possible triples of vertices $\{x_u, x'_u, x_{v^i}\}$ and we chose the triplet corresponding to the quadratic system with the most accurate solutions. Therefore, BP verifies whether either or both positions x_{v^0}, x_{v^1} are compatible with the distances to all pairs of adjacent predecessors, by selecting the best one solving (4.11). We notice that here BP needs a common pair of precedents, and not necessarily the *immediate* predecessors as required for solving DMDGP instances.

In our implementation, we solve the quadratic system (4.11) by Gaussian Elimination [45], for which two linear systems need to be solved. Since the distances between pairs of vertices in $\{x_u, x'_u, x_{v^i}\}$ can be large, the coordinates related to such vertices may have distinct orders of magnitude. This can cause the occurrence of badly-scaled matrices for the two systems to be solved. Therefore, we employed the function `dgesvx` of the LAPACK library [6], which automatically scales the coefficient matrices before solving the linear systems.

4.5.3 Pruning the branches of the tree

If there exist other vertices in $N(v) \cap \gamma(v)$ which have not been used to compute I, then they can be used for checking the feasibility of the two computed positions. BP prunes the branch x_v, together with all the positions along the same branch, only if the corresponding distances are infeasibles with respect to (4.1), for a given tolerance $\varepsilon > 0$, i.e., if

$$|||x_u - x_v|| - d_{u,v}| > \varepsilon \quad \forall u \in N(v) \cap \gamma(v) \wedge u \notin U_v. \tag{4.12}$$

This pruning device is called *Direct Distance Feasibility* (DDF), and it is one of the most robust pruning devices that can be used during the discrete search.

Another pruning device, based on the point-to-point Dijkstra shortest path, searches on Euclidean graphs [92]. Consider the vertices u, v, w with $u < v < w$ such that $\{u, w\} \in E$, i.e., the distance $d_{u,w}$ is known. Suppose that a position for the vertex u is already available, and that the feasibility of the node x_v needs to be verified. Let $D(v, w)$ be an upper bound to the distance $||x_v - x_w||$ for all possible valid embeddings. Then, if

$$||x_u - x_v|| > d_{u,w} + D(v, w) \tag{4.13}$$

holds, the node x_v can be pruned [92] because the triangular inequality is negated. A valid upper bound $D(v, w)$ can be computed by finding the shortest path between the vertex v and the vertex w in G. We call this pruning device *Dijkstra Shortest Path* (DSP). Computational experiments showed that the SDP detects infeasible embeddings sooner than the DDF, but it is also more computationally expensive.

In the case of the DMDGP, new pruning devices were proposed in [106] in order to consider real NMR experiments. In fact, it has been noticed that the range of distances between the hydrogen atoms pairs of the molecule is very large, and often DDF is not able to sufficiently prune branches of the tree. This causes the multiplication of the solutions found by BP, where some infeasible solutions are also contained. Such new pruning devices, instead, are based on different information, such as the list of bounds on the torsion angles of the protein backbones, and the information regarding the protein secondary structures.

4.5.4 Algorithm accuracy evaluation

The BP algorithm supplies in output: (i) an embedding x, (ii) the CPU time employed by the method to yield x, and (iii) an assessment about how far is x from a known optimal solution. For this purpose can be used several accuracy measures, most popular are the following.

1. *penalty*, which is the evaluation of the function defined in (4.8) for a given embedding x.

2. *Largest Distance Error* (LDE):

$$LDE = \frac{1}{m} \sum_{\{u,v\} \in E} \frac{|\,||x_u - x_v|| - d_{u,v}|}{d_{u,v}}$$

   it is a scaled, averaged and square-rooted version of the penalty.

3. *Root Mean Square Deviation* (RMSD), which is a difference measure for sets of points in Euclidean space having the same center of mass. Let x, y be embeddings of G, then

$$RMSD(x, y) = \min_{T} ||y - Tv||$$

   where T varies over all rotations and translations in $\mathbb{R}^k$.

4. *Max Error* (ME), which is the maximum distance between the position found and the real position of the vertex v. This is defined as

$$ME = \max_v ||x_v - x_v^{real}||_2.$$

Accordingly, if y is the known optimal configuration of a given protein, different realizations of the same protein yield different RMSD values. Evidently, RMSD is a meaningful accuracy measure only for test sets where the optimal conformations are already known (such as PDB instances). We, for our part, have chosen to use the LDE function. Theoretically, if the value of the LDE is 0 then x is feasible and (4.1) holds; in practice, the value of the LDE has to be lesser then a very small tolerance parameter $\varepsilon > 0$.

4.6 Computational complexity

The DGP can be solved in polynomial time if the distances for all pairs of vertices are available [74]. Nevertheless, it has been proved to be a strongly NP-Complete for $k = 1$ and strongly NP-hard for $k > 1$ by reduction from SUBSET-SUM problem [118]. In the same manner in [94] has been shown the NP-hardness of the DMDGP. In [9] the sensors network is defined as an unit disk graph, i.e., an intersection graphs composed by circumferences centered in the vertices and which edges are Euclidean distances at most twice the radius. The authors showed that finding an embedding in $\mathbb{R}^2$ for a unit disk graph of given radius is still a NP-hard problem.

Despite this NP-hardness complexity, the scientific community has not been discouraged from searching and developing new algorithms for solving the DGP, because the theory related to NP-hardness of the problem given in [118] was based on very special graphs and distances, which are highly unlikely to occur in practical problems on proteins. In fact, by exploiting the natural atomic ordering of these molecules in various way we can produce a vertex order as described in Section 4.4.3. As mentioned in Section 4.5, BP has in the worst-case an exponential running time. Nevertheless, reordering the instances BP is able to find all the solutions of the problem in polynomial time [94].

4.7 Computational Results

We present in this section some computational experiments related to the two BP versions for the MDGP and the SNLP presented in [89] and [51] respectively. In both cases, before to run BP was checked whether all instances satisfied the discretization assumptions. In presence of a problem whose vertex order did not satisfied the discretization

assumptions the instance were reordered as described in Section 4.4.3. In all the experiments, the tolerance used when comparing known and computed distances was set to $\varepsilon = 0.001$. The code used for running all tests was written in C programming language. All tests were done on a 2.13GHz Intel Core 2 Duo processor and with 4GB of RAM, running Linux. The codes have been compiled by the GNU C compiler 4.1.2 version with the `-O3` flag.

4.7.1 MDGP experiments

We considered instances generated from proteins having known conformations in the PDB [19]. Each PDB record consisted in a set of atomic coordinates for a given protein generated by computing the distances between all the possible pairs of hydrogens in the molecule, and by keeping only the ones smaller than a predefined threshold δ. This procedure simulated NMR data, because all the distances were between hydrogens, and only short-range distances were considered. The threshold δ usually ranges between 5Å[3] and 6Å, thus here was set $\delta = 5.5$Å.

For each protein, in Table 4.1 there is the number of its hydrogen atoms, and the number of available distances (edges). The next two columns refer to the BP algorithm applied to the reordered proteins. We report the solution quality in terms of LDE and the user CPU time, expressed in seconds. The last two columns refer to DGSOL [103], a software for distance geometry based on a continuous formulation of the problem.

Because DGSOL accepts a set of lower and upper bounds on the available distances as input (and therefore solves a different problem than ours), the comparison is not totally fair. Moreover, DGSOL is the only well-known continuous optimization-based algorithm with publicly available code that we could use as a reference to compare against. DGSOL was provided of the set of intervals $[d - \varepsilon, d + \varepsilon]$, where d is the generic distance given to BP. The obtained values for the LDE function and the user CPU time show that BP is faster (by around 2 orders of magnitude) and able to find better-quality (by around 10 orders of magnitude) solutions.

4.7.2 SNLP experiments

If the solution of the optimization problem is not conditioned from the noise, the algorithm can return correct sensor positions [7]. Therefore, for all instances we assumed that there was not noise in the distance measurements.

[3] Å is the unit of length for intermolecular distances. One angstrom equals to 10^{-10} meters, or 0.1 nanometers.

Instance	Atoms	|E|	BP LDE	BP time	DGSOL LDE	DGSOL time
1brv	90	729	3.36e-11	0.01	4.14e-01	1.90
1a11	144	1192	2.43e-12	0.01	1.07e-05	5.27
1erp	209	1969	3.63e-11	0.05	3.95e-01	7.21
1aqr	214	1690	3.45e-11	0.02	6.19e-01	8.34
1bbl	221	1690	2.19e-08	0.05	9.29e-01	9.81
1ed7	261	2591	3.91e-11	0.05	8.34e-01	8.04
1h1j	261	2489	3.16e-11	0.03	3.41e-01	13.08
1ahl	268	2508	4.33e-11	0.02	6.46e-01	15.03
1dv0	275	2669	4.08e-10	0.05	9.20e-01	14.47
1k1v	277	2600	4.25e-11	0.06	7.42e-01	12.66
1ccq	389	3888	5.97e-11	0.10	7.47e-01	20.46
1a2s	480	4723	5.71e-08	0.77	7.72e-01	24.75
1acz	589	6067	5.36e-08	1.97	7.42e-01	44.15
2hsy	620	5935	8.23e-11	0.66	8.10e-01	32.66
1b4c	1152	11044	7.62e-08	1.81	9.22e-01	117.51
1a23	1157	11628	9.08e-11	2.38	8.79e-01	110.00
2ron	1501	15101	1.09e-06	4.15	8.47e-01	148.61
1ezo	2259	21049	4.89e-07	7.91	9.09e-01	308.90

TABLE 4.1: Comparison between BP and DGSOL performances on a set of molecular instances.

The instances were composed of n sensors (including anchors) randomly placed in the region $[0, 1]^2$. We ranged the number of sensors from 2000 to 10000 in steps of 2000, and the radio range R from 0.7 to 0.4 in steps of 0.1. We also generated the partial Euclidean distance matrix of order n such that its no unspecified elements were the distance between anchor pairs and the distance between sensors that are within R.

In Table 4.2 we report the results of our tests on noiseless problems. The first two columns show the cardinally of the vertex set (number of sensors plus number of anchors) and the cardinally of the edge set. Note that the first 4 vertices were taken as anchors, as specified in the third column. The other columns report the value of the radio range R, the best LDE function values and the CPU time (seconds) needed to solve the instance with BP. The last two columns refer to the performance of the SNLS-DPclique (here referred as SDPcl) facial reduction algorithm presented in [85]. This mixed-combinatorial algorithm identifies the intersections of faces of the SDP cone as unions of intersecting cliques and iteratively expand them following a vertex order.

The comparison between the BP and the SDPcl algorithm shows that although the latter scales extremely well, BP is faster with a slightly lower order of accuracy of the solution: $O(10^{12})$ for BP versus $O(10^{13})$ for SDPcl.

Sensors	|E|	Anchors	R	BP LDE	BP time	SDPcl ME	SDPcl time
2000	28892	4	0.07	1.30e-12	0.16	6e-13	1.00
2000	21490	4	0.06	1.16e-12	0.17	1e-12	1.00
2000	14634	4	0.05	9.90e-13	0.15	-	1.00
4000	116490	4	0.07	1.33e-12	0.57	2e-13	2.00
4000	86325	4	0.06	1.40e-12	0.60	6e-13	2.00
4000	59991	4	0.05	1.45e-12	0.61	4e-13	2.00
4000	38807	4	0.04	4.33e-12	0.58	1e-13	2.00
6000	261120	4	0.07	7.96e-13	1.09	3e-13	4.00
6000	194750	4	0.06	1.27e-12	1.39	2e-13	4.00
6000	135334	4	0.05	8.91e-8	1.30	3e-13	3.00
6000	87715	4	0.04	2.38e-12	1.38	7e-13	3.00
8000	464562	4	0.07	8.35e-13	2.08	3e-13	6.00
8000	344007	4	0.06	1.35e-12	2.40	2e-13	6.00
8000	241254	4	0.05	2.52e-12	2.97	6e-13	5.00
8000	154699	4	0.04	1.77e-12	2.70	6e-13	5.00
10000	721838	4	0.07	9.24e-13	3.70	3e-13	9.00
10000	536027	4	0.06	9.75e-13	3.67	9e-13	8.00
10000	377435	4	0.05	1.98e-12	3.97	5e-13	7.00
10000	242951	4	0.04	1.76e-12	4.51	3e-13	7.00

TABLE 4.2: Comparison between BP and SDPcl performances on a set of noiseless problems.

4.8 Conclusions

In this chapter we introduced the MDGP and the SNLP as subclasses of instances of the DGP. After a brief review of the existing approaches for solving the DGP, we described few combinatorial requirements necessaries to reduce the search space from continuous to discrete, by defining the DDGP. We explained the needed requirements for discretizing also MDGP and SNLP instances in order to apply the BP algorithm proposed in [94, 97]. In such papers BP is used to solve only DMDGP instances, but we adapted it in a later stage to solve also SNLP instances [51]. In order to test the efficacy of BP in both problems, we reported the computational results obtained comparing BP with two well-know algorithms. For the molecular case, we reported the comparison results between BP and DGSOL [103] on a set of molecular sparse instance not previously ordered. These instances were artificially generated from a subset of proteins having known conformations in the PDB [19]. For the sensor networks case, we compared BP and SDPcl [85] on a set of instances generated with same parameters.

The results highlight the importance of the discretization of the solution space, by finding faster an optimal solution without loss of quality. On the other hand, algorithms working on Euclidean spaces are often iterative on the graph vertices, and therefore require a vertex order. For this purpose, we have shown that a vertex re-ordering might

transform an instance that did not belong to the class of the DDGP to one that instead satisfied the assumptions of discretization. Comparing the average CPU time needed to reorder a DMDGP instance and a DSNLP instance, we observed that the reordering algorithm was very fast only with MDGP instances. In fact, the longest molecular chain represented by a graph with $n = 2259$ and $|E| = 21049$ was reordered in less than 0.20 [89]. This could be imputed to two factors: (i) the larger cardinality of the vertex set in DSNLP instances (the smallest had $|V| = 2000$), and (ii) a better spatial conformation of the graph for molecular instances given by the protein backbone shape.

In many real cases, the distances are not provided as exact values but they are given in estimated ranges (e.g., in protein modeling the distances are often provided with some estimated bounds), by involving the not uniquely of the solution. For this purpose, we are working on a new extension the BP algorithm in order to manage this kind of data. In this framework we refer to the iBP algorithm, presented in [91] for solving MDGPs, which is is the first algorithm implementing a discrete search which is able to manage interval data. In particular, we are carrying out the development of new discretization orders for the amino acids that can be found in proteins. Together with the discretization order for the protein backbone described in Section 4.4, these orders allow for discretizing all instances concerning proteins composed by these amino acids. In this way, the employment of the iBP algorithm is possible, and all distances estimated through NMR experiments can be exploited for efficiently pruning parts of the iBP search tree.

In conclusion, future researches about the MDGP will be devoted to the identification of special orders for the side chains in proteins. The aim is to improve the iBP algorithm, so that it can output more representative solutions. In the framework of the wireless sensor networks, instead, we will complete the corresponding iBP version and we will focus on the improvement of the reordering procedure, which is crucial to obtain a remarkable speed-up of the algorithm in order to test larger instances.

Chapter 5

Conclusions

5.1 Summary

In this dissertation we studied two combinatorial problems related to the molecular structure determination. Knowing the three-dimensional structures of RNA and proteins is essential to understand their biological functions. Unfortunately, these structures are very difficult to determine for various technical reasons [18]. The two main experimental techniques used to derive structure models of biological biomolecules are X-ray crystallography and Nuclear Magnetic Resonance spectroscopy. X-ray crystallography uses diffraction data of a molecule crystal to find the electron density distribution of the molecule, and hence its structure; whereas the magnetic resonance spectra of the nuclear spins in a molecule can be detected by NMR spectroscopy and used to estimate the distances between certain pairs of atoms and subsequently, the coordinates of the atoms in the molecule. In either case, a set of experimental data is collected and a mathematical problem needs to be solved to form the structure [119]. X-ray crystallography outdoes NMR in the resolution of experimental data. Nevertheless, the crystallization of several RNA molecules is not possible, it does not reflect the molecular dynamics, and often it is not reproduce the real conformation of the molecules [124, 129]. High resolution NMR study can provide both structural details and dynamic characteristics of the molecule. For these reasons, NMR is a powerful tool for the analysis of folding transitions in molecules. The process of molecular structure determination by NMR data consists of a sequence of steps: data acquisition and processing, peak picking, assignment, derivation of spatial restraints, structure calculation, and validation [70, 124, 129]. The two problems discussed in this dissertation cover two steps of this process.

The first topic is the *orderly colored longest path problem for RNA structure determination*. We described the problem of assignment pathway reconstruction of a RNA

molecule by 3D NMR maps. The idea is that determining the sequence of interactions among atoms involved in the NMR experiment can lead to determine the shape of such biological molecule. We proved that assignment pathway problem is NP-hard, and showed a formulation based on edge-colored graphs. Taking into account that interactions between consecutive nuclei in the NMR spectrum are different according to the type of residue along the RNA chain, each color in the graph represented a type of interaction. Thus, we represented the sequence of interactions as the problem of finding a longest (hamiltonian in the best case) path under the constraint that the edges of the path followed a given order of colors. We defined this problem as the Orderly Colored Longest Path problem on a c-edge-colored graph (OCLP). We also showed how the OCLP problem can be adapted to model different types of real problems. Next, we described three alternative IP formulations used to solve the OCLP problem, which differ in the way the orderly colored paths over a directed network. In order to prove the efficacy of this approach, we tested our models over two sets of randomly generated problems with different characteristics. Moreover, we proposed a second instance generator that takes into account the biological component information of the problem. In fact, assignment of cross-peaks on the 3D NMR maps recorded for RNA molecules is a relatively new, and not experimental data collected for already solved cases are available for an extended analysis.
The first set of experiments concerned instances generated as edge-colored graph on which paths of 2 or 3 colors were sought. The comparison among the three formulations in the first set of experiments has shown that two models, where cycle separation was performed iteratively adding constraints to a lighter formulation, performed much better. For this reason we considered a second set of experiments focused only on these two models. We developed a Branch & Cut procedure that uses a cycle separation polynomial oracle in order to enumerate all optimal solutions for the majority of the considered problems on a standard computer. The computational results of the second set of experiments confirmed that the proposed models were valid options to solve 3D-APP problems of realistic sizes. In particular, most of large size instances (100 cross-peaks) have been solved within the time bound of one hour.

Once resonance signals have been identified, it is possible to calculate some *structural restraints* of the molecule, such as molecular distances (e.g., distances between hydrogen atoms), torsion angles (dihedral angles around certain bonds), coupling constants, etc. [142]. The most popular methods used for structure generation are distance geometry (DG) methods developed in order to satisfy the covalent and structural restraints [90]. The atoms coordinates provide a structure that can be considered as an approximation to the real one [74]. This structure can be refined by using optimization, e.g., by an energy minimization procedure with the distance ranges as structural restraints. Here

takes place the second topic of the thesis, the *discretizable distance geometry problem* (DDGP). This is a revision of the above mentioned distance geometry problem (DGP), which, in addition to protein modeling, has applications in many other fields from statistics to engineering. After a brief review of the existing approaches for solving the DGP, we showed few combinatorial requirements necessaries to reduce the search space from continuous to discrete, by defining the DDGP. We focused on two particular applications of this problem, one in protein modeling and one in sensor networks localization. Thus, we defined the discretizable molecular distance geometry problem (DMDGP) and the discretizable sensor network localization problem (DSNLP) respectively. We described the Branch & Prune algorithm proposed in [94, 97] for molecular instances, and a new version of this combinatorial algorithm that we developed in order to solve sensor network instances. In order to test the efficacy of BP in both problems, we have shown the computational results obtained comparing BP with two well-know algorithms. For the molecular case, we reported the comparison results between BP and DGSOL [103] on a set of molecular sparse instance not previously ordered. These instances were artificially generated from a subset of proteins having known conformations in the PDB [19]. For the sensor networks case, we compared BP and SDPcl [85] on a set of instances generated with same parameters.

The results highlight the importance of the discretization of the solution space, by finding faster an optimal solution without loss of quality. On the other hand, algorithms working on Euclidean spaces are often iterative on the graph vertices, and therefore require a vertex order. For this purpose, we have shown that a vertex re-ordering might transform an instance that did not belong to the class of the DDGP to one that instead satisfied the assumptions of discretization. Comparing the average CPU time needed to reorder a DMDGP instance and a DSNLP instance, we observed that the reordering algorithm was very fast only with MDGP instances. In fact, the longest molecular chain represented by a graph with $n = 2259$ and $|E| = 21049$ was reordered in less than 0.20 [89]. This could be imputed to two factors: (i) the larger cardinality of the vertex set in DSNLP instances (the smallest had $|V| = 2000$), and (ii) a better spatial conformation of the graph for molecular instances given by the protein backbone shape.

5.2 Ongoing work and Future directions

As regards the OCLP problem for RNA structure determination, we are working on the extension of the method to large real experimental data as well as to improve the spectra generator, in order to make it also useful to test alternative methods. We are developing some theoretical conjectures (i.e., about the presence of OCLP (or OCHP) in the graph related to some its characteristics), and improving the formulations and

the whole assignment procedure. Moreover, it is our intention to implement a dynamic programming algorithm working in presence of cycles, in order to solve an instance without call the commercial solvers.

Future researches on MDGP will be devoted to the identification of special orders for the side chains in proteins. In many real cases, indeed, the distances are only provided for the protein backbone. Moreover, such distances are often not provided as exact values but they are given in estimated ranges (e.g., in protein modeling the distances are often provided with some estimated bounds), by involving the not uniquely of the solution. For this purpose, we are working on a new extension the BP algorithm in order to manage this kind of data. In this framework we refer to the iBP algorithm, presented in [91] for solving MDGPs, which is is the first algorithm implementing a discrete search which is able to manage interval data. Together with the discretization order for the protein backbone described in this thesis, this new ordering allows to discretize all instances concerning proteins also composed by amino acids. In this way, all distances estimated through NMR experiments can be exploited for efficiently pruning parts of the iBP search tree, and it can output more representative solutions. In the framework of the wireless sensor networks, instead, we will complete the corresponding iBP version and we will focus on the improvement of the reordering procedure, which is crucial to obtain a remarkable speed-up of the algorithm in order to test larger instances.

Appendix A

Algebra background

A.1 Vectors and Matrices

In this thesis, we work with vectors and matrices whose components are real numbers. Vectors are denoted by lowercase letters, and matrices by uppercase letters. The space of real vectors of length n is denoted by $\mathbb{R}^n$, and the space of real $m \times n$ matrices is denoted by $\mathbb{R}^{m \times n}$.

Given a vector $x \in \mathbb{R}^n$, we use x_i to denote its ith component, and assume that x is a column vector such that its transpose, denoted by x^T, is a row vector, i.e. $x = (x_1, \cdots, x_n)^T$. The product between a vector x and a scalar α is $\alpha x = (\alpha x_1, \cdots, \alpha x_n)^T$. If $x, y \in \mathbb{R}^n$, the standard *inner product* is

$$xy = x^T y = \sum_{i=1}^{n} x_i y_i.$$

Given a matrix $A \in \mathbb{R}^n$, we specify its components by double scripts as $a_{ij}, i = 1, \cdots, m$ and $j = 1, \cdots, n$. Let $x \in \mathbb{R}^n$ be a vector and let $A \in \mathbb{R}^n$ be a matrix. Then the *matrix-vector product* $b = Ax$ is the m-dimensional column vector defined as follows:

$$b_i = \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \cdots, m.$$

The *transpose* of an $m \times n$ matrix A, denoted by A^T, is the $n \times m$ matrix such that $a_{ij}^T = a_{ji}$, $i = 1, \cdots, n$ and $j = 1, \cdots, m$. A matrix A is said to be *square* if $m = n$. A square matrix A is *symmetric* if $A^T = A$. The diagonal of the matrix $A \in \mathbb{R}^n$ consists of the elements a_{ii}, for $i = 1, \cdots, min(m, n)$. A is called *diagonal* if $a_{ij} = 0$ whenever $i \neq j$. The *identity* matrix, denoted by I, is the square diagonal matrix whose diagonal elements are all 1.

A square matrix $A \in \mathbb{R}^n$ is called *nonsingular* if there exists an $n \times n$ matrix B such that $AB = BA = I$. We denote B by A^{-1} and call it the *inverse* of A. For a nonsingular matrix $A \in \mathbb{R}^n$ and for any vector $b \in \mathbb{R}^n$, there exists $x \in \mathbb{R}^n$ such that $Ax = b$.
A set of n vectors $x_i, \cdots, x_n$ is *linearly independents* if none of them can be written as linear combination of the others, that is iff does not exist n scalars $\alpha_1, \cdots, \alpha_n$ not all zero such that

$$\sum_{i=1}^{n} \alpha_i x_i = 0.$$

If such scalars exist, then the vectors $x_i, \cdots, x_n$ are said to be *linearly dependents*.

A.2 Norms

A *norm* is a mapping $||\cdot||$ from $\mathbb{R}^n$ to the nonnegative real numbers that satisfies the following:

- $||x|| = 0 \Leftrightarrow x = 0$ for all $x \in \mathbb{R}^n$,
- $||\alpha x|| = |\alpha|\ ||x||$ for all $\alpha \in \mathbb{R}$ and $x \in \mathbb{R}^n$,
- $||x + y|| \leq ||x|| + ||y||$ for all $x, y \in \mathbb{R}^n$.

For any vector $x \in \mathbb{R}^n$, one can define the following norms:

- $||x||_1 = \sum_{i=1}^{n} |x_i|$,
- $||x||_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2}$,
- $||x||_\infty = \max_{i=1,\cdots,n} |x_i|$.

The norm $||\cdot||_2$ is often called the *Euclidean* norm, and it satisfies the Cauchy-Schwarz inequality

$$||x^T y|| \leq ||x||_2 ||y||_2,$$

with equality if and only if one of these vectors is a nonnegative multiple of the other.

A.3 Gaussian Elimination

Let $A \in \mathbb{R}^n$ be the square matrix whose columns are the vectors $a_i, i = 1, \cdots, n$. If the vectors $a_i, \cdots, a_n$ are linearly independent then this matrix is nonsingular. Hence the nonlinear equations

$$||x - a_i||_2^2 = d_i^2, \quad i = 1, \cdots, n,$$

or equivalently,

$$x^T x - 2x^T a_i + a_i^T a_i = d_i^2, \quad i = 1, \cdots, n,$$

can be rewrite as

$$a_i^T x = (r + b_i)/2, \quad i = 1, \cdots, n \tag{A.1}$$

where $r = x^T x$ and $b_i = a_i^T a_i - d_i^2$, $i = 1, \cdots, n$.
In matrix form (A.1) become $A^T x = (re + b)/2$, or

$$x = (ru + v)/2, \tag{A.2}$$

where $e \in \mathbb{R}^n$ denotes the vector $e = [1, 1, \cdots, 1]^T$ and

$$u = A^{-T} e, \qquad v = A^{-T} b. \tag{A.3}$$

Hence, $r = x^T x = \frac{1}{4}(ru + v)^T (ru + v)$ or $(u^T u)r^2 + (2u^T v - 4)r + v^T v = 0$, which is a quadratic equation in the scalar r. Solving for r gives,

$$r = \frac{2 - u^T v \pm \sqrt{(2 - u^T v)^2 - (u^T u)(v^T v)}}{u^T u}, \tag{A.4}$$

and the two solutions for x can then be recovered using (A.2).

The above approach is efficient, requiring the solution of two linear systems of equations (A.3) of order n [45].

Appendix B

Simplex volumes and the Cayley-Menger determinant

B.1 The simplex volumes

A simplex in n-dimensional Euclidean space is a convex solid with $n+1$ vertices. Thus in one-dimensional space a simplex is just the line segment between two specified points. A simplex in two-dimensional space is a triangle (three vertices), a simplex in three-dimensional space is a tetrahedron (four vertices), and so on. Hence, any vertex of a simplex in a k-dimensional space can be regarded as the apex of a pyramid on a (k-1)-dimensional base defined by the other vertices.
The *content* of a simplex (i.e., the length of a one-dimensional simplex, the area of a two-dimensional simplex, the volume of a three-dimensional simplex, and so on) can be expressed very simply as a function of the coordinates of the $n+1$ vertices.
Let V_{n-1} denote the content of the base, the content V_n of the pyramid is given by

$$V_n = \frac{1}{n!} \prod_{i=1}^{n} h_i, \tag{B.1}$$

where h_1 is the distance between the first two vertices, h_2 is the height of the third vertex above the line containing those two vertices, h_3 is the height of the fourth vertex above the plane containing the first three vertices, and so on. Thus the content of a n-dimensional simplex is $1/n!$ times the heights of the vertices (taken in any linear sequence) above the subspace containing the previous vertices.

In matrix terms, we can rewrite (B.1) as the determinant of an $n \times n$ matrix consisting of the n coordinates of each of the remaining n vertices. For example, with $n = 3$ the content of the corresponding 3-simplex, with the fourth vertex located at the origin, is

given by

$$V_3 = \frac{1}{3!}\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix},$$

where (x_i, y_i, z_i) with $i = 1, 2, 3$ are the coordinates of the other three vertices.
We can also write the volume of the simplex without requiring one of them to be at the origin, but considering a fourth vertex (x_4, y_4, z_4). The content of this 3-simplex is

$$V_3 = \frac{1}{3!}\begin{bmatrix} x_1 - x_4 & y_1 - x_4 & z_1 - x_4 \\ x_2 - x_4 & y_2 - x_4 & z_2 - x_4 \\ x_3 - x_4 & y_3 - x_4 & z_3 - x_4 \end{bmatrix}.$$

By co-factor decomposition we can write this as the determinant of an $n+2$ dimensional matrix

$$V_3 = \frac{1}{3!}\begin{bmatrix} 1 & x_1 - x_4 & y_1 - x_4 & z_1 - x_4 \\ 1 & x_2 - x_4 & y_2 - x_4 & z_2 - x_4 \\ 1 & x_3 - x_4 & y_3 - x_4 & z_3 - x_4 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

We can make use of the fact that adding a multiple of any column (or row) to another column (or row) does not change the determinant. In fact, if any column vector c_i is replaced with $c_i + kc_j$, where c_j is one of the other column vectors, then each co-factor is multiplied by an element of $c_i + kc_j$, and hence it is clear that the determinant is the sum of the original determinant plus the determinant of the original matrix where c_i has been replaced by kc_j. But the latter determinant is identically zero, because one column is a multiple of another. Therefore, applying this proposition to the above matrix, we can add x_4 times the first column to the second column, obtaining:

$$V_3 = \frac{1}{3!}\begin{bmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & x_4 & x_4 \end{bmatrix}.$$

This derivation is completely general, and applied to simplexes in n dimensions, enables us to compute the content in terms of the coefficients of the $(n + 1)$ vertices.

B.2 The Cayley-Menger determinant

In the thesis we express the volume of a simplex in terms of the edge lengths rather than the vertex coordinates. To illustrate how we can derive such an expression, consider a

two-dimensional simplex with vertices $p_i = (x_i, y_i)$, $i = 1, 2, 3$. We know the content is given by the determinant

$$V_2 = \frac{1}{2!} \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}.$$

Since a matrix and its transpose have the same determinant, we also have

$$(V_2)^2 = \left(\frac{1}{2} \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \right) \left(\frac{1}{2} \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_3 & y_3 \end{bmatrix} \right) = \frac{1}{4} \begin{bmatrix} 1 + p_1 p_1 & 1 + p_1 p_2 & 1 + p_1 p_3 \\ 1 + p_2 p_1 & 1 + p_2 p_2 & 1 + p_2 p_3 \\ 1 + p_3 p_1 & 1 + p_3 p_2 & 1 + p_3 p_3 \end{bmatrix},$$

where $p_i p_j = x_i x_j + y_i y_j$. We can express this as the determinant of a matrix of dimension increased by one, as follows.

$$(V_2)^2 = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 + p_1 p_1 & 1 + p_1 p_2 & 1 + p_1 p_3 \\ 0 & 1 + p_2 p_1 & 1 + p_2 p_2 & 1 + p_2 p_3 \\ 0 & 1 + p_3 p_1 & 1 + p_3 p_2 & 1 + p_3 p_3 \end{bmatrix}.$$

We again make use of the fact that the determinant of a matrix is unchanged if any multiple of a row is added to any other row. Thus we can subtract the first row from each of the other rows to give

$$(V_2)^2 = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & p_1 p_1 & p_1 p_2 & p_1 p_3 \\ -1 & p_2 p_1 & p_2 p_2 & p_2 p_3 \\ -1 & p_3 p_1 & p_3 p_2 & p_3 p_3 \end{bmatrix}.$$

Let'us notice that the determinant of the co-factor of the upper left element is zero, as can be seen from the fact that

$$\begin{bmatrix} 0 & x_1 & y_1 \\ 0 & x_2 & y_2 \\ 0 & x_3 & y_3 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} = \begin{bmatrix} p_1 p_1 & p_1 p_2 & p_1 p_3 \\ p_2 p_1 & p_2 p_2 & p_2 p_3 \\ p_3 p_1 & p_3 p_2 & p_3 p_3 \end{bmatrix}.$$

The determinants on the left are obviously zero, so the right side is also zero. Hence the upper-left element of the prior matrix has no effect on the value of the determinant, so we can set it to zero. Also, making use of the fact that multiplying the elements of any column (or row) by a constant has the effect of multiplying the determinant by that

constant, we can negate the first column to give

$$(V_2)^2 = -\frac{1}{4}\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & p_1p_1 & p_1p_2 & p_1p_3 \\ 1 & p_2p_1 & p_2p_2 & p_2p_3 \\ 1 & p_3p_1 & p_3p_2 & p_3p_3 \end{bmatrix}.$$

If we multiply each of the last three columns by -2, and then multiply the first row by -1/2, this expression becomes

$$(V_2)^2 = -\frac{1}{16}\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & -2p_1p_1 & -2p_1p_2 & -2p_1p_3 \\ 1 & -2p_2p_1 & -2p_2p_2 & -2p_2p_3 \\ 1 & -2p_3p_1 & -2p_3p_2 & -2p_3p_3 \end{bmatrix}.$$

Now, for $i = 1, 2, 3$, we add the first column multiplied by p_ip_i to the $(i+1)$th column, and we add the first row multiplied by p_ip_i to the $(i+1)$th row, to give the expression

$$(V_2)^2 = -\frac{1}{16}\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & p_1^2 - 2p_1p_1 + p_1^2 & p_1^2 - 2p_1p_2 + p_2^2 & p_1^2 - 2p_1p_3 + p_3^2 \\ 1 & p_2^2 - 2p_2p_1 + p_1^2 & p_2^2 - 2p_2p_2 + p_2^2 & p_2^2 - 2p_2p_3 + p_3^2 \\ 1 & p_3^2 - 2p_3p_1 + p_1^2 & p_3^2 - 2p_3p_2 + p_2^2 & p_3^2 - 2p_3p_3 + p_3^2 \end{bmatrix}.$$

The square of the distance between p_i and p_j is $d_{i,j}^2 = p_i^2 - 2p_ip_j + p_j^2 = (x_i - x_j)^2 + (y_i - y_j)^2$. Thus, we compute the Cayley-Menger determinant for the area of a triangle in terms of the edge lengths as

$$(V_2)^2 = -\frac{1}{16}\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & d_{1,2}^2 & d_{1,3}^2 \\ 1 & d_{2,1}^2 & 0 & d_{2,3}^2 \\ 1 & d_{3,1}^2 & d_{3,2}^2 & 0 \end{bmatrix}.$$

Therefore, the Cayley-Menger determinant giving the squared content of an n-dimensional simplex can be written as

$$(V_n)^2 = -\frac{|d_{i,j}^2|}{(-2)^n(n!)^2}.$$

where $i, j = 0, 1, \cdots, n+1$.

Thus the squared content of a one-dimensional simplex (i.e., a line segment between two

vertices) can trivially be expressed in terms of the edge length as

$$2(V_1)^2 = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & d_{1,2}^2 \\ 1 & d_{2,1}^2 & 0 \end{bmatrix} = d_{1,2}^2 + d_{2,1}^2 = 2d_{1,2}^2$$

Likewise the volume of a tetrahedron is given in terms of the edge lengths by

$$(V_3)^2 = -\frac{1}{16} \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & d_{1,2}^2 & d_{1,3}^2 & d_{1,4}^2 \\ 1 & d_{2,1}^2 & 0 & d_{2,3}^2 & d_{2,4}^2 \\ 1 & d_{3,1}^2 & d_{3,2}^2 & 0 & d_{3,4}^2 \\ 1 & d_{4,1}^2 & d_{4,2}^2 & d_{4,3}^2 & 0 \end{bmatrix}.$$

Bibliography

[1] Abouelaoualim A., K.Ch. Das, L. Faria, Y. Manoussakis, C. Martinhon, R. Saad, *Paths and trails in edge-colored graphs*, Theoretical Computer Science, 409: 497-510, 2008.

[2] Aggarwal A., A. Bar-Noy, D. Coppersmith, R. Ramaswani, B. Schieber, M. Sudan, *Efficient routing and scheduling algorithms for optical networks*, In Proc. 5th ACM-SIAM SODA, 1994.

[3] Ahuja R.K., T.L. Magnanti, J.B. Orlin, *Network Flows: Theory, Algorithms and Application*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[4] Albert B., A. Johnson, J. Lewis, M. Raff, K. Roberts, P. Walter, *Molecular Biology of the cell*, 4th edition, N.Y., Garland Science, 2002.

[5] Alon N., J. Grytczuk, M. Haluszczak, O. Riordan, *Non-repetitive colorings of graphs*, Random Structure and Algorithms, 21(3-4): 336-346, 2002.

[6] Anderson E., Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. Du Croz, S. Hammerling, J. Demmel, C. Bischof, D. Sorensen, *LAPACK: a Portable Linear Algebra Library for High-Performance Computers*, In Proc. of the 1990 ACM/IEEE conference on supercomputing, 2-11, IEEE Computer Society Press, 1990.

[7] Anderson B.D.O., I. Shames, G. Mao, B. Fidan, *Formal Theory of Noisy Sensor Network Localization*, SIAM J. Discrete Math., 24(2): 684-698, 2010.

[8] Apaydin M.S., B. Çatay, N. Patrick, B.R. Donald, *NVR-BIP: Nuclear Vector Replacement Using Binary Integer Programming for NMR Structure-Based Assignments*, The Computer J., 54(5): 708-716, 2011.

[9] Aspnes J., T. Eren, D.K. Goldenberg, A.S. Morse, W. Whiteley, Y.R. Yang, B.D.O. Anderson, P.N. Belhumeur, *A theory of network localization*, IEEE Transactions on Mobile Computing, 12: 1663-1678, 2006.

[10] Bachrach J., C. Taylor, *Localization in Sensor Networks*, in Handbook of Sensor Networks: Algorithms and Architectures, I. Stojmenović (Eds.), Wiley, New York, 277-310, 2005.

[11] Bahrami A., A. Assadi, J.L. Markley, H. Eghbalnia, *Probabilistic Interaction Network of Evidence Algorithm and its Application to Complete Labeling of Peak lists from Protein NMR Spectroscopy*, PLoS Computat Biol., 5(3): e1000307, 2009.

[12] Bahrami A., L.J. Clos II, J.L. Markley, S.E. Butcher, H.R. Eghbalnia, *RNA-PAIRS: RNA probabilistic assignment of imino resonance shifts*, J Biomol NMR., 52(4): 289302, 2012.

[13] Banga J.R., *Optimization in computational systems biology*, BMC Systems Biology, 2:47, 2008.

[14] Bang-Jensen J., G. Gutin, *Alternating cycles and trails in in 2-edge-colored complete multigraphs*, Discrete Math., 188: 61-72, 1998.

[15] Bang-Jensen J., G. Gutin, *Properly colored Hamiltonian paths in edge-coloured complete graphs*, Discrete Appl. Math., 82: 247-250, 1998.

[16] Bartel D.P., *MicroRNAs: target recognition and regulatory functions*, Cell, 136(2): 215-33, 2009.

[17] Benkouar A., Y. Manoussakis, V. Th. Paschos and R. Saad, *On the complexity of some Hamiltonian and Eulerian problems in edge-colored complete graphs*, RAIRO-Operations Research 30: 417-438, 1996.

[18] Berg J.M., J.L. Tymoczko, L. Stryer, *Biochemistry*, W. H. Freeman, 2006.

[19] Berman H., J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. Shindyalov, P. Bourne, *The protein data bank*, Nucleic Acid Research 28: 235-242, 2000.

[20] Bertolazzi P., G. Felici, P. Festa, *Logic Based Methods for SNPs Tagging and Reconstruction*, Computers & Operations Research, 37: 1419-1426, 2010.

[21] Bertsekas D.P., *Nonlinear Programming: 2nd edition*, Athena Scientific, 1999.

[22] Biswas P., T. Lian, T. Wang, Y. Ye, *Semidefinite Programming based algorithms for Sensor Network Localization*, ACM Transactions on Sensor Networks, 2: 188-220, 2006.

[23] Blazewicz J., P. Formanowicz, M. Kasprzak, *Selected combinatorial problems of computational biology*, European Journal of Operational Research, 161: 585-597, 2005.

[24] Blazewicz J., M. Szachniuk, A. Wojtowicz, *RNA tertiary structure determination: NOE pathways construction by tabu search*, Bioinformatics, 21: 2356-2361, 2005.

[25] Blumenthal L., *Theory and Applications of Distance Geometry*, Oxford University Press, Oxford, 1953.

[26] Bohr H., S. Brunak, *Protein structure by distance analysis*, IOS Press, 1994.

[27] Bollobás B., *Graph Theory. An Introductory Course*, Springer-Verlag, New york, 1979.

[28] Bollobás B., P. Erdos, *Properly edge colored hamiltonian cycles*, Israel Journal of Mathematics, 23: 126-131, 1976.

[29] Bondy J.A., U.S.R. Murty, *Graph Theory with Applications*, Macmillan Press, London, 1976.

[30] Bracewell R.N., *The Fourier Transform and its Applications*, third edition, McGraw-Hill, Boston, 2000.

[31] Brady T., C. Watt, *On products of Euclidean reflections*, American Mathematical Monthly, 113: 826-829, 2006.

[32] Brandes U., C. Pich, *An Experimental Study on Distance-Based Graph Drawing*, Graph Drawing, Lecture Notes in Computer Science, 5417: 218-229, 2009.

[33] Broersma H., X. Li, *Spanning trees with many or few colors in edge-colored graphs*, Discus. Math. Graph Theory, 17: 259-269, 1997.

[34] Broersma H., X. Li, S. Zhang, *Paths and cycles in colored graphs*, Electr. Notes in Discr. Math., 8, 2001.

[35] Bulterman R.W., W. van der Sommen, G. Zwaan, T. Verhoeff, A.J.M. van Gasteren, W.H.J. Feijen, *On computing a longest path in a tree*, Inform. Process. Lett., 81: 93-96, 2002.

[36] Carrabs F., R. Cerulli, M. Gentili, *The labeled maximum matching problem*, Computers & Operations Research, 36(6): 1859-1871, 2009.

[37] Carter M.W., H.H. Jin, M.A. Saunders, Y. Ye, *SPASELOC: an adaptive subproblem algorithm for scalable wireless Sensor Network Localization*, Journal on Optimization, 17(4): 1102-1128, 2006.

[38] Cavanagh J., W.J. Fairbrother, A.G. Palmer, N.J. Skelton, *Protein NMR Spectroscopy: Principals and Practice*, Academic Press, 2006.

[39] Cerulli R., P. Dell'Olmo, M. Gentili, A. Raiconi, *Heuristic approaches for the Minimum Labelling Hamiltonian Cycle Problem*, Electronic Notes in Doscrete Mathematics, 25(1): 131-138, 2006.

[40] Cerulli R., A. Fink, M. Gentili, S. Voss, *Metaheuristics comparison for the minimum labelling spanning tree problem*, in B.L. Golden, S. Raghavan and E.A. Wasil (eds.), The Next Wave on Computing, Optimization, and Decision Technologies, Springer, New York, 93-106, 2005.

[41] Chang R.S., S. Leu, *The minimum labeling spanning trees*, Information Processing Letters, 63(5): 277-282, 1997.

[42] Chen L., R.-S. Whang, X.-S. Zhang, *Biomolecular networks: methods and applications in system biology*, Wiley series in bioinformatics, 2009.

[43] Chou W.S., Y. Manoussakis, O. Megalakaki, M. Spyratos, Zs. Tuza, *Paths through fixed vertices in edge-colored graphs*, Mathmatiques et sciences humaines, 127: 49-58, 1994.

[44] Cook S.A., *The complexity of theorem-proving procedures*, in Proc. of 3rd Annual ACM Symp. on Theory of Comp., N.Y., 151-158, 1971.

[45] Coope L.D., *Reliable computation of the points of intersection of n spheres in* R^n, ANZIAM Journal, 42: C461-C477, 2000.

[46] Crick F., *On protein Synthesis*, Symp. Soc. Exp. Biol., 12: 138-63, 1958.

[47] Crippen G.M., T.F. Havel, *Distance Geometry and molecular conformation*, John Wiley & Sons, New York, 1988.

[48] Dantzig G.B., *Linear Programming and Extensions*, Princeton University Press, Princeton, 1963.

[49] De Cola M.C., G. Felici, M. Szachniuk, *The Orderly Colored Longest Path Problem*, CNR-IASI Technical Report, 29/2012.

[50] De Cola M.C., G. Felici, D. Santoni, E. Weitschek, *Filtering with alignment free distances for high throughput DNA reads assembly*, EMBnet.journal, North America, 18, 2012.

[51] De Cola M.C., A. Mucherino, G. Felici, L. Liberti, *A Branch and Prune algorithm for the Sensor Networks Localization Problem*, CNR-IASI Technical Report, R.13-04, 2013.

[52] Dong Q., Z. Wu, *A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data*, Journal of Global Optimization, 26: 321-333, 2003.

[53] Dongarra J., F. Sullivan, *Computing in Science and Engineering*, 2(1): 22-23, 2000.

[54] Dorninger D., *On permutations of chromosomes*, In Contributions of General Algebra, 5: 95-103, 1987.

[55] Dorninger D., *Hamiltonian circuits determining the order of chromosomes*, Discrete Appl. Math., 50: 159-168, 1994.

[56] Drexl M., I. Stefan, *Solving elementary shortest path problems as mixed-integer programs*, OR Spectrum, 1-16, 2012.

[57] Eren T., D.K. Goldenberg, W. Whiteley, Y.R. Yang, A.S. Morse, B.D.O. Anderson, P.N. Belhumeur, *Rigidity, computation, and randomization in network localization*, IEEE Infocom Proc., 2673-2684, 2004.

[58] Feillet D., P. Dejax, M. Gendreau, C. Gueguen, *An exact algorithm for the elementary shortest path problem with resource constarints: application to some vehicle routing problems*, Networks, 44(3): 216-229, 2004.

[59] Fellows M.R., J. Guo, I.A. Kanj, *The parameterized complexity of some minimum label problems*, Journal of Computer System Science, 76: 727-740, 2010.

[60] Festa P., *The shortest path tour problem: problem definition, modeling, and optimization*, in Proc. of INOC'2009, 1-7, 2009.

[61] Fischetti M., *Lezioni di Ricerca Operativa*, Edizioni Libreria Progetto Padova, second edition, 1999.

[62] Gallo J.M., P. Jin, C.A. Thornton, H. Lin, J. Robertson, I. D'Souza, W.W. Schlaepfer, *The role of RNA and RNA processing in neurodegeneration*, Journal of Neuroscience, 25: 10372-10375, 2005.

[63] Gallo G., S. Pallottino, *Shortest path methods: A unifying approach*, Math. Programming Studies, 26: 38-64, 1986.

[64] Gandham S., M. Dawande, R. Prakash, *Link scheduling in wireless sensor networks: distributed edge-coloring revisited*, Journal of Parallel and Distributed computing, 68(8): 1122-1134, 2008.

[65] Garey M.R., D.S. Johnson, *Computers and Intractability: A guide to the Theory of NP-Completeness*, W.H. Freeman & Co., San Francisco CA, 1979.

[66] Gilbert W., *The RNA World*, Nature 319: 618-619, 1986.

[67] Greenberg H.J., W.E. Hart, G. Lancia, *Opportunities for combinatorial optimization in computational biology*, INFORMS Journal on Computing, 16(3): 211-231, 2004.

[68] Golub G.H., C.F. van Loan, *Matrix Computations*, Johns Hopkins University Press, 1989.

[69] Gourvès L., A. Lyra, C. Martinhon, J. Monnot, F. Protti, *On s-t paths and trails in edge-colored graphs*, Electronic Notes in Discrete Math., 35: 221-226, 2009.

[70] Günter P., *Automated structure determination from NMR spectra*, European Biophysics Journal, 38(2): 129-143, 2009.

[71] Gutin G., *Finding a Longest path in a complete multipartite digraph*, SIAM Journal Discrete Math., 6: 270-273, 1993.

[72] Gutin G., B. Sudakov, A. Yeo, *Note on alternating directed cycles*, Discrete Math., 191: 101-107, 1998.

[73] Harel D., Y. Koen, *Graph Drawing by High-Dimensional Embedding*, Journal of Graph Algorithms and Applications, 8(2): 195-214, 2004.

[74] Havel T.F., *Distance Geometry*, in Encyclopedia of Nuclear Magnetic Resonance, D.M. Grant and R.K. Harris (Eds.), Wiley, New York, 1701-1710, 1995.

[75] He L., J.M. Thomson, M.T. Hemann, E. Hernando-Monge, D. Mu, S. Goodson, S. Powers, C. Cordon-Cardo, S.W. Lowe, G.J. Hannon, S.M. Hammond, *A microRNA polycistron as a potential human oncogene*, Nature 435(7043): 828-33, 2005.

[76] Hendrickson B., *The Molecular Problem: Determining Conformation from Pairwise Distances*, Ph.D. thesis, Cornell University, 1991.

[77] Herrmann T., P. Güntert, K. Wüthrich, *Protein NMR structure determination with automated NOE assignment using the new software CANDID and the torsion angle dynamics algorithm DYANA*, Journal of molecular biology, 319: 209-227, 2002.

[78] Hesse M., H. Meier, B. Zeeh., *Metodi spettroscopici nella chimica organica*, Edises, 1996.

[79] Ioannidou K., G.B. Mertzios, S.D. Nikolopoulos, *The longest path problem has a polynomial solution on interval graphs*, Algorithmica, 61: 320-341, 2011.

[80] IBM, *ILOG-CPLEX 12.2 user's manual*, IBM, 2012.

[81] Kaklamanis C., P. Persiano, *Efficient wavelength routing on directed fiber trees*, in Proc. of 4th ESA'96, Lecture Notes in Computer Science, 1136: 460-470, 1996.

[82] Karger D.R., R. Motwani, G.D.S. Ramkumar, *On approximating the longest path in a graph*, Algorithmica, 18: 82-98, 1997.

[83] Karush W., *Minima of Functions of Several Variables with Inequalities as Side Constraints*, Masters thesis, Department of Mathematics, University of Chicago, 1939.

[84] Kirby M., *Operational Research in War and Peace: The British Experience from the 1930s to 1970*, Imperial College Press, London, 2003.

[85] Krislock N., H. Wolkowicz, *Explicit sensor network localization using semidefinite representations and facial redeuction*, SIAM Journal on Optimization, 20(2): 679-2708, 2010.

[86] Kuhn H.W., A. W. Tucker, *Nonlinear Programming*, In J. Neyman (Eds.), Proc. of the 2nd Berkeley Symp. on Math. Statistics and Prob., University of California Press, Berkeley, CA, USA, 481-492, 1950.

[87] Lancia G., *Mathematical Programming in Computational Biology: an annotated Bibliography*, *Algorithms*, 1(2): 100-129, 2008.

[88] Larrañaga P., B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J.A. Lozano, R. Armañanzas, G. Santafé, A. Pérez, V. Robles, *Machine learning in bioinformatics*, Briefings in bioinformatics, 7(1): 86-112, 2006.

[89] Lavor C., J. Lee, A. Lee-St. John, L. Liberti, A. Mucherino, M. Sviridenko, *Discretization orders for distance geometry problems*, Optimization Letters, 6: 783-796, 2012.

[90] Lavor C., L. Liberti, N. Maculan, *Molecular Distance Geometry Problem*, in Encyclopedia of Optimization, C. Floudas and P. Pardalos (Eds.), 2305-2311, Springer, New York, second edition, 2009.

[91] Lavor C., L. Liberti, A. Mucherino, *The interval Branch-and-Prune Algorithm for the Discretizable Molecular Distance Geometry Problem with Inexact Distances*, Journal of Global Optimization, 56(3): 855-871, 2013.

[92] Lavor C., L. Liberti, A. Mucherino, N. Maculan, *On a discretizable subclass of instances of the molecular distance geometry problem*, in D. Shin (Eds), Proc. of the 24th Annual ACM Symp. on Appl. Comp., 804-805, 2009.

[93] Li H., G. Wang, S. Zhou, *Long alternating cycles in edge-colored complete graphs*, Lecture Notes in Computer Science, 4613: 305-309, 2007.

[94] Liberti L., C. Lavor, N. Maculan, *A Branch-and-Prune algorithm for the Molecular Distance Geometry Problem*, Technical Report q-bio/0608012, arXiv, 2006.

[95] Liberti L., C. Lavor, N. Maculan, *A branch-and-prune algorithm for the molecular distance geometry problem*, International Transactions in Operational Research, 15:1-17, 2008.

[96] Liberti L., C. Lavor, A. Mucherino, N. Maculan, *Molecular Distance Geometry Methods: from continuous to discrete*, International Transactions in Operational Research, 18:33-51, 2010.

[97] Liberti L., B. Masson, J. Lee, C. Lavor, A. Mucherino, *On the number of solutions of the Discretizable Molecular Distance Geometry Problem*, in Combinatorial Optimization, COCOA11, 6831: 322-342, 2011.

[98] Linge J.P., M. Habeck, W. Rieping, M. Nilges, *ARIA: automated NOE assignment and NMR structure calculation*, Bioinformatics, 19: 315-316, 2003.

[99] Lukasiak P., M. Antczak, T. Ratajczak, J.M. Bujnicki, M. Szachniuk, M. Popenda, R.W. Adamiak, J. Blazewicz, *RNAlyzer - novel approach for quality analysis of RNA structural models*, Nucleic Acids Research 41(12): 5978-90, 2013.

[100] Manoussakis Y., *Alternating paths in edge-colored complete graphs*, Discrete Mathematics, 56: 297-309, 1995.

[101] Mertzios G.B., D.G. Corneil, *A simple polynomial algorithm for the longest path problem on cocomparability graphs*, Discrete Math., 26(3): 940-963, 2012.

[102] Moseley H.N.B., D. Monleon, G.T. Montelione, *Automatic determination of protein backbone resonance assignments from triple-resonance NMR data*, Methods in Enzymology 339: 91-108, 2001.

[103] Moré J.J., Z. Wu, *Global continuation for distance geometry problems*, SIAM J. Optim., 7: 814-836, 1997.

[104] Mraz M., S. Pospisilova, *MicroRNAs in chronic lymphocytic leukemia: From causality to associations and back*, Expert Review of Hematology 5(6): 579-581, 2012.

[105] Mucherino A., C. Lavor, L. Liberti, *The Discretizable Distance Geometry Problem*, Optimization Letters, 6(8): 1671-1686, 2012.

[106] Mucherino A., C. Lavor, T. Malliavin, L. Liberti, M. Nilges, N. Maculan, *Influence of pruning devices on the solution of Molecular Distance Geometry Problems*, Lecture Notes in Computer Science, 6630: 206-217, 2011.

[107] Mucherino A., L. Liberti, C. Lavor, *MD-jeep: an Implementation of a Branch & Prune Algorithm for Distance Geometry Problems*, Lectures Notes in Computer Science, in Proc. of ICMS10, Kobe, Japan, 6327: 186-197, 2010.

[108] Myers B.R., *Enumeration of tour in Hamiltonian rectangular Lattice graphs*, Mathematical Magazine, 54: 19-23, 1981.

[109] Pal A., *Localization algorithms in Wireless Sensor Networks: current approaches and future challenges*, Network Protocols and Algorithms, 2(1), 2010.

[110] Papadimitriou C.H., K. Steiglitz, *Combinatorial optimization: algorithms and complexity*, Dover Pubblications, Inc., Mineola, NY, USA, 1998.

[111] Petitjean M., *Sphere unions and intersections and some of their applications in molecular modeling*, in Distance Geometry: Theory, Methods, and Applications, A. Mucherino, C. Lavor, L. Liberti, and N. Maculan (Eds), Springer, Berlin, 2013.

[112] Pevzner P.A., *Computational Molecular Biology: an algorithmic approach*, MIT Press, 2000.

[113] Pevzner P.A., *DNA Physical mapping and alternating eulerian cycles in colored graphs*, Algorithmica, 13(1-2): 77-105, 1995.

[114] Raghavan P., E. Upfal, *Efficient routing in all optical networks*, in Proc. of 26th ACM STOC, 1994.

[115] Restrepo G., J.L. Villaveces, *Mathematical thinking in chemistry*, Hyle: International Journal for Philosophy of Chemistry, 18: 3-22, 2012.

[116] Sachenbacher M., M. Leucker, A. Artmeier, J. Haselmayr, *Efficient Energy-Optimal Routing for Electric Vehicles*, in Proc. of AAAI, 2011.

[117] Savelsbergh M.W.P., M. Sol, *The general pickup and delivery problem*, Transport Sci, 29: 17-29, 1995.

[118] Saxe J.B., *Embeddability of weighted graphs in k-space is strongly Np-hard*, in Proc. of 17th Allerton Conference in Communications, Conrol and Computing, Monticello, 480-489, 1979.

[119] Schlick T., *Molecular Modeling and Simulation: An Interdisciplinary Guide*, Springer, 2003.

[120] Schmidt E., P. Güntert, *A new algorithm for reliable and general NMR resonance assignment*, J. Am. Chem. Soc., 134: 12817-12829, 2012.

[121] Selkoe D.J., *Folding proteins in fatal ways*, Nature 426(6968): 900-904, 2003.

[122] Sippl M., H. Scheraga, *Cayley-Menger coordinates*, Proc. Natl. Acad. Sci. USA, 83: 2283-2287, 1986.

[123] Sit A., Z. Wu, *Solving a generalized distance geometry problem for protein structure determination*, Bullettin of Mathematical Biology, 73: 2809-2836, 2011.

[124] Spronk C.A.E.M., S.B. Nabuurs, E. Krieger, G. Vriend, G.W. Vuister, *Validation of protein structures derived by NMR spectroscopy*, Progress in Nuclear Magnetic Resonance Spectroscopy, 45: 315-337, 2004.

[125] Sum M., Xiong M., *A mathematical programming approach for gene selection and tissue classification*, Bioinformatics, 19(10): 1243-51, 2003.

[126] SylvesterJ., *Chemistry and algebra*, Nature, 17: 284-284, 1878.

[127] Szachniuk M., M.C. De Cola, G. Felici, D. de Werra, J. Blazewicz, *Optimal pathway reconstruction on 3D NMR maps*, Discrete Applied Mathematics, in revision.

[128] Szachniuk M., M.C. De Cola, G. Felici, J. Blazewicz, *The Orderly Colored Longest Path Problem - A survey of applications and new algorithm*, RAIRO-Operations Research, accepted.

[129] Szachniuk M., M. Popenda, R.W. Adamiak, J. Blazewicz, *An Assignment Walk through 3D NMR Spectrum*, in Proc. of CIBCB'2009, 215-219, 2009.

[130] Szeider S., *Finding paths in graphs avoiding forbidden transitions*, Discrete Appl. Math., 126: 239-251, 2003.

[131] Tseng I.-L., H.-W. Chen, C.-I. Lee, *Obstacle-Aware longest path routing with parallel MILP solvers*, in Proc. of WCECS, 2010.

[132] Tubaishat M., S. Madria, *Sensor networks: an overview*, in IEEE Potentials, 22: 20-23, 2003.

[133] Uehara R., Y. Uno, *Effcient algorithms for the longest path problem*, in Proc. of 15th ISAAC, 871-883, 2004.

[134] Uehara R., G. Valiente, *Linear structure of bipartite permutation graphs and the longest path problem*, Inform. Process. Lett., 103: 71-77, 2007.

[135] Ulrich E.L., H. Akutsu, J.F. Doreleijers, Y. Harano, Y.E. Ioannidis, J. Lin, M. Livny, S. Mading, D. Maziuk, Z. Miller, E. Nakatani, C.F. Schulte, D.E. Tolmie, R.K. Wenger, H. Yao, J.L. Markley, *BioMagResBank*, Nucleic Acids Research 36(1): D402-D408, 2008.

[136] Watkins J.J., R.L. Hoenigman, *Knight's tours on a torus*, Mathematics, 70: 175-184, 1997.

[137] Weber G.-W., S. Özöğür, E. Kropat, *A Review on data mining and continuous optimization applications in computational biology and medicine*, Birth Defect Research (Part C), 87: 165-181, 2009.

[138] Weitschel E., G. Felici, P. Bertolazzi, *MALA: A Microarray clustering and classification software*, Wagner RR, Hameurlain A, Tjoa AM (Eds.), 24th SSDBM, 2012.

[139] Williamson D.P., L.A. Hall, J.A. Hoogeveen, C.A.J. Hurkens, J.K. Lenstra, S.V. Sevast'janov, D.B. Shmoys, *Short Shop Schedules*, Operations Research, 45(2): 288-294, 1997.

[140] Wolsey L.A., *Integer Programming*, John Wiley & Sons, 1998.

[141] Wu D., Z. Wu, *An updated geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data*, Journal of Global Optimization, 37: 661-673, 2007.

[142] Wüthrich K., *NMR of Proteins and Nucleic Acids*, John Willey & Sons, New York, 1986.

[143] Xiong Y., B. Golden, E. Wasil, *The Colorful Traveling Salesman problem*, in Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies, 37: 115-123, 2007.

[144] Yeo A., *A note on alternating cycles in edge-colored graphs*, Journal of Combinatorial Theory, Series B-69: 222-225, 1997.

[145] Yoon J., Y. Gad, Z. Wu, *Mathematical modeling of protein structure with Distance Geometry*, Numerical Linear Algebra and Optimization, Scientific Press, 2002.

[146] Zimmerman D.E., C.A. Kulikowski, Y. Huang, W. Feng, M. Tashiro, S. Shimotakahara, C-Y. Chien, R. Powers, G.T. Montelione, *Automated analysis of protein NMR assignments using methods from artificial intelligence*, Journal of Molecular Biology, 269: 592-610, 1997.